

e-puck

1.0

Generated by Doxygen 1.8.11



# Contents

- 1 e-puck standard library documentation 1**
  - 1.1 Introduction . . . . . 1
  - 1.2 Documentation organization . . . . . 1
  - 1.3 External links . . . . . 2
  
- 2 Module Index 3**
  - 2.1 Modules . . . . . 3
  
- 3 Data Structure Index 5**
  - 3.1 Data Structures . . . . . 5
  
- 4 File Index 7**
  - 4.1 File List . . . . . 7
  
- 5 Module Documentation 11**
  - 5.1 Analogic/Digital conversion (ADC) . . . . . 11
    - 5.1.1 Detailed Description . . . . . 12
    - 5.1.2 Introduction . . . . . 12
    - 5.1.3 Package organization . . . . . 12
  - 5.2 Bluetooth . . . . . 13
    - 5.2.1 Detailed Description . . . . . 13
    - 5.2.2 Introduction . . . . . 13
  - 5.3 Camera fast two timers . . . . . 14
    - 5.3.1 Detailed Description . . . . . 14
    - 5.3.2 Introduction . . . . . 14

---

5.3.3	Default settings	14
5.3.4	Performances	15
5.3.5	Important note	15
5.3.6	Examples	15
5.3.6.1	Basic example	15
5.3.6.2	More advanced example	16
5.3.7	Introduction	16
5.4	Camera slow three timers	17
5.4.1	Detailed Description	17
5.4.2	Introduction	17
5.4.3	Default settings	17
5.4.4	Performances	18
5.4.5	Exemples	18
5.4.5.1	Basic example	18
5.4.5.2	More advanced example	18
5.5	Sound	19
5.5.1	Detailed Description	19
5.5.2	Introduction	19
5.5.3	Package organization	19
5.5.4	Sounds plage	20
5.6	LIS sensor turret	21
5.6.1	Detailed Description	21
5.6.2	Introduction	21
5.7	Radio communication	22
5.7.1	Detailed Description	22
5.7.2	Introduction	22
5.8	FFT	23
5.8.1	Detailed Description	23
5.8.2	Introduction	23
5.8.3	How it works	23

---

5.9	I2C	24
5.9.1	Detailed Description	24
5.9.2	Introduction	24
5.9.3	Overview of I2C protocol	24
5.9.3.1	Master-slave relation	25
5.9.3.2	Start and Stop conditions	25
5.9.3.3	Transferring data	25
5.9.4	Use I2C on e-puck	26
5.9.4.1	Basics functions	26
5.9.4.2	More developed functions	26
5.9.5	Reference	26
5.10	Matlab communication	27
5.10.1	Detailed Description	27
5.10.2	Introduction	27
5.10.2.1	Sending data from matlab	27
5.10.2.2	Sending data from e-puck	27
5.11	Ports, motors and LEDs	28
5.11.1	Detailed Description	29
5.11.2	Introduction	29
5.11.2.1	Ports	29
5.11.2.2	Motors	29
5.11.2.3	LED	29
5.11.2.4	IR remote	29
5.11.3	Timer's problems	30
5.11.4	Agenda solution	30
5.12	UART	31
5.12.1	Detailed Description	31
5.12.2	Introduction	31
5.12.3	Bluetooth	31

---

<b>6</b>	<b>Data Structure Documentation</b>	<b>33</b>
6.1	AgendaList Struct Reference	33
6.1.1	Detailed Description	33
6.1.2	Field Documentation	33
6.1.2.1	agendas	33
6.1.2.2	motors	33
6.1.2.3	speed	33
6.1.2.4	timers_in_use	34
6.1.2.5	waiting	34
6.2	AgendaType Struct Reference	34
6.2.1	Detailed Description	34
6.2.2	Field Documentation	34
6.2.2.1	activate	34
6.2.2.2	counter	34
6.2.2.3	cycle	35
6.2.2.4	function	35
6.2.2.5	next	35
6.3	BtDevice Struct Reference	35
6.3.1	Detailed Description	35
6.3.2	Field Documentation	35
6.3.2.1	address	35
6.3.2.2	class	36
6.3.2.3	friendly_name	36
6.4	BtEPuck Struct Reference	36
6.4.1	Detailed Description	36
6.4.2	Field Documentation	36
6.4.2.1	address	36
6.4.2.2	number	36
6.5	TypeAccRaw Struct Reference	37
6.5.1	Detailed Description	37
6.5.2	Field Documentation	37
6.5.2.1	acc_x	37
6.5.2.2	acc_y	37
6.5.2.3	acc_z	37
6.6	TypeAccSpheric Struct Reference	37
6.6.1	Detailed Description	38
6.6.2	Field Documentation	38
6.6.2.1	acceleration	38
6.6.2.2	inclination	38
6.6.2.3	orientation	38

---

<b>7 File Documentation</b>	<b>39</b>
7.1 a_d/advance_ad_scan/e_acc.c File Reference	39
7.1.1 Detailed Description	40
7.1.2 Function Documentation	41
7.1.2.1 calculate_acc_raw(void)	41
7.1.2.2 calculate_acc_spherical(void)	41
7.1.2.3 e_acc_calibr(void)	41
7.1.2.4 e_display_angle(void)	41
7.1.2.5 e_get_acc(unsigned int captor)	41
7.1.2.6 e_get_acc_filtered(unsigned int captor, unsigned int filter_size)	42
7.1.2.7 e_read_acc(void)	42
7.1.2.8 e_read_acc_spheric(void)	42
7.1.2.9 e_read_acc_x(void)	42
7.1.2.10 e_read_acc_xyz(void)	43
7.1.2.11 e_read_acc_y(void)	43
7.1.2.12 e_read_acc_z(void)	43
7.1.2.13 e_read_inclination(void)	43
7.1.2.14 e_read_orientation(void)	43
7.1.3 Variable Documentation	44
7.1.3.1 acc_x	44
7.1.3.2 acc_y	44
7.1.3.3 acc_z	44
7.1.3.4 acceleration	44
7.1.3.5 angle_mem	44
7.1.3.6 centre_x	44
7.1.3.7 centre_y	44
7.1.3.8 centre_z	44
7.1.3.9 e_acc_scan	44
7.1.3.10 e_last_acc_scan_id	44
7.1.3.11 inclination	44

7.1.3.12	isCalibratingFlag	44
7.1.3.13	lastAccX	44
7.1.3.14	lastAccY	44
7.1.3.15	lastAccZ	44
7.1.3.16	orientation	44
7.1.3.17	updateAccl2CCounter	44
7.2	a_d/advance_ad_scan/e_acc.h File Reference	44
7.2.1	Detailed Description	46
7.2.2	Macro Definition Documentation	46
7.2.2.1	ACCX_BUFFER	46
7.2.2.2	ACCY_BUFFER	46
7.2.2.3	ACCZ_BUFFER	46
7.2.2.4	ANGLE_ERROR	46
7.2.2.5	CST_RADIAN	46
7.2.2.6	FILTER_SIZE	47
7.2.2.7	GRAVITY	47
7.2.2.8	GRAVITY_LSM330	47
7.2.3	Function Documentation	47
7.2.3.1	e_acc_calibr(void)	47
7.2.3.2	e_display_angle(void)	47
7.2.3.3	e_get_acc(unsigned int captor)	47
7.2.3.4	e_get_acc_filtered(unsigned int captor, unsigned int filter_size)	47
7.2.3.5	e_read_acc(void)	48
7.2.3.6	e_read_acc_spheric(void)	48
7.2.3.7	e_read_acc_x(void)	48
7.2.3.8	e_read_acc_xyz(void)	48
7.2.3.9	e_read_acc_y(void)	48
7.2.3.10	e_read_acc_z(void)	49
7.2.3.11	e_read_inclination(void)	49
7.2.3.12	e_read_orientation(void)	49



7.3	<a href="#">a_d/advance_ad_scan/e_ad_conv.c File Reference</a>	49
7.3.1	<a href="#">Detailed Description</a>	50
7.3.2	<a href="#">Function Documentation</a>	50
7.3.2.1	<a href="#">__attribute__((__interrupt__, auto_psv))</a>	50
7.3.2.2	<a href="#">e_ad_is_acquisition_completed(void)</a>	50
7.3.2.3	<a href="#">e_ad_is_array_filled(void)</a>	51
7.3.2.4	<a href="#">e_ad_scan_off(void)</a>	51
7.3.2.5	<a href="#">e_ad_scan_on(void)</a>	51
7.3.2.6	<a href="#">e_init_ad_scan(unsigned char only_micro)</a>	51
7.3.3	<a href="#">Variable Documentation</a>	51
7.3.3.1	<a href="#">centre_z</a>	51
7.3.3.2	<a href="#">e_acc_scan</a>	51
7.3.3.3	<a href="#">e_ambient_and_reflected_ir</a>	51
7.3.3.4	<a href="#">e_ambient_ir</a>	51
7.3.3.5	<a href="#">e_last_acc_scan_id</a>	52
7.3.3.6	<a href="#">e_last_mic_scan_id</a>	52
7.3.3.7	<a href="#">e_mic_scan</a>	52
7.3.3.8	<a href="#">is_ad_acquisition_completed</a>	52
7.3.3.9	<a href="#">is_ad_array_filled</a>	52
7.3.3.10	<a href="#">micro_only</a>	52
7.3.3.11	<a href="#">selector</a>	52
7.3.3.12	<a href="#">tickAdclsr</a>	52
7.3.3.13	<a href="#">updateAccl2CCounter</a>	52
7.4	<a href="#">a_d/e_ad_conv.c File Reference</a>	52
7.4.1	<a href="#">Detailed Description</a>	52
7.4.2	<a href="#">Function Documentation</a>	52
7.4.2.1	<a href="#">e_init_ad(void)</a>	52
7.4.2.2	<a href="#">e_read_ad(unsigned int channel)</a>	52
7.5	<a href="#">a_d/advance_ad_scan/e_ad_conv.h File Reference</a>	53
7.5.1	<a href="#">Detailed Description</a>	54

7.5.2	Macro Definition Documentation	54
7.5.2.1	ACC_PROX_PERIOD	54
7.5.2.2	ACC_PROX_SAMP_FREQ	54
7.5.2.3	ACC_SAMP_NB	54
7.5.2.4	ADC_ISR_PERIOD_MS	54
7.5.2.5	ADCS_2_CHAN	54
7.5.2.6	ADCS_3_CHAN	54
7.5.2.7	ADCS_5_CHAN	54
7.5.2.8	ADCS_6_CHAN	54
7.5.2.9	ALL_ADC	55
7.5.2.10	MIC_SAMP_FREQ	55
7.5.2.11	MIC_SAMP_NB	55
7.5.2.12	MICRO_ONLY	55
7.5.2.13	PULSE_LENIGHT	55
7.5.2.14	PULSE_PERIOD	55
7.5.3	Function Documentation	55
7.5.3.1	e_ad_is_acquisition_completed(void)	55
7.5.3.2	e_ad_is_array_filled(void)	55
7.5.3.3	e_ad_scan_off(void)	55
7.5.3.4	e_ad_scan_on(void)	55
7.5.3.5	e_init_ad_scan(unsigned char only_micro)	55
7.6	a_d/e_ad_conv.h File Reference	56
7.6.1	Detailed Description	56
7.6.2	Function Documentation	56
7.6.2.1	e_init_ad(void)	56
7.6.2.2	e_read_ad(unsigned int channel)	56
7.7	a_d/advance_ad_scan/e_micro.c File Reference	57
7.7.1	Detailed Description	57
7.7.2	Function Documentation	57
7.7.2.1	e_get_micro(unsigned int micro_id)	57

---

7.7.2.2	<a href="#">e_get_micro_average(unsigned int micro_id, unsigned int filter_size)</a>	58
7.7.2.3	<a href="#">e_get_micro_last_values(int micro_id, int *result, unsigned samples_nb)</a>	58
7.7.2.4	<a href="#">e_get_micro_volume(unsigned int micro_id)</a>	58
7.7.3	<a href="#">Variable Documentation</a>	59
7.7.3.1	<a href="#">e_last_mic_scan_id</a>	59
7.7.3.2	<a href="#">e_mic_scan</a>	59
7.8	<a href="#">a_d/e_micro.c File Reference</a>	59
7.8.1	<a href="#">Detailed Description</a>	59
7.8.2	<a href="#">Function Documentation</a>	60
7.8.2.1	<a href="#">e_get_micro(int *m0, int *m1, int *m2)</a>	60
7.8.2.2	<a href="#">e_init_micro(void)</a>	60
7.9	<a href="#">a_d/advance_ad_scan/e_micro.h File Reference</a>	60
7.9.1	<a href="#">Detailed Description</a>	61
7.9.2	<a href="#">Macro Definition Documentation</a>	61
7.9.2.1	<a href="#">MIC0_BUFFER</a>	61
7.9.2.2	<a href="#">MIC1_BUFFER</a>	61
7.9.2.3	<a href="#">MIC2_BUFFER</a>	61
7.9.3	<a href="#">Function Documentation</a>	61
7.9.3.1	<a href="#">e_get_micro(unsigned int micro_id)</a>	61
7.9.3.2	<a href="#">e_get_micro_average(unsigned int micro_id, unsigned int filter_size)</a>	61
7.9.3.3	<a href="#">e_get_micro_volume(unsigned int micro_id)</a>	62
7.10	<a href="#">a_d/e_micro.h File Reference</a>	62
7.10.1	<a href="#">Detailed Description</a>	62
7.10.2	<a href="#">Function Documentation</a>	63
7.10.2.1	<a href="#">e_get_micro(int *m1, int *m2, int *m3)</a>	63
7.10.2.2	<a href="#">e_init_micro(void)</a>	63
7.11	<a href="#">a_d/advance_ad_scan/e_prox.c File Reference</a>	63
7.11.1	<a href="#">Detailed Description</a>	64
7.11.2	<a href="#">Function Documentation</a>	64
7.11.2.1	<a href="#">e_calibrate_ir()</a>	64

7.11.2.2	<code>e_get_ambient_light(unsigned int sensor_number)</code> . . . . .	64
7.11.2.3	<code>e_get_calibrated_prox(unsigned int sensor_number)</code> . . . . .	65
7.11.2.4	<code>e_get_prox(unsigned int sensor_number)</code> . . . . .	65
7.11.3	Variable Documentation . . . . .	66
7.11.3.1	<code>e_ambient_and_reflected_ir</code> . . . . .	66
7.11.3.2	<code>e_ambient_ir</code> . . . . .	66
7.11.3.3	<code>init_value_ir</code> . . . . .	66
7.12	<code>a_d/e_prox.c</code> File Reference . . . . .	66
7.12.1	Detailed Description . . . . .	67
7.12.2	Function Documentation . . . . .	67
7.12.2.1	<code>__attribute__((interrupt, auto_psv))</code> . . . . .	67
7.12.2.2	<code>e_get_ambient_light(unsigned int sensor_number)</code> . . . . .	67
7.12.2.3	<code>e_get_prox(unsigned int sensor_number)</code> . . . . .	68
7.12.2.4	<code>e_init_prox(void)</code> . . . . .	68
7.12.2.5	<code>e_stop_prox(void)</code> . . . . .	68
7.12.2.6	<code>init_tmr1(void)</code> . . . . .	69
7.12.3	Variable Documentation . . . . .	69
7.12.3.1	<code>ambient_and_reflected_ir</code> . . . . .	69
7.12.3.2	<code>ambient_ir</code> . . . . .	69
7.12.3.3	<code>reflected_ir</code> . . . . .	69
7.13	<code>a_d/advance_ad_scan/e_prox.h</code> File Reference . . . . .	69
7.13.1	Detailed Description . . . . .	69
7.13.2	Function Documentation . . . . .	70
7.13.2.1	<code>e_calibrate_ir()</code> . . . . .	70
7.13.2.2	<code>e_get_ambient_light(unsigned int sensor_number)</code> . . . . .	70
7.13.2.3	<code>e_get_calibrated_prox(unsigned int sensor_number)</code> . . . . .	70
7.13.2.4	<code>e_get_prox(unsigned int sensor_number)</code> . . . . .	71
7.14	<code>a_d/e_prox.h</code> File Reference . . . . .	72
7.14.1	Detailed Description . . . . .	72
7.14.2	Function Documentation . . . . .	73

7.14.2.1	<a href="#">e_get_ambient_light(unsigned int sensor_number)</a>	73
7.14.2.2	<a href="#">e_get_prox(unsigned int sensor_number)</a>	73
7.14.2.3	<a href="#">e_init_prox(void)</a>	74
7.14.2.4	<a href="#">e_stop_prox(void)</a>	74
7.15	<a href="#">a_d/e_accelerometer.c File Reference</a>	75
7.15.1	<a href="#">Detailed Description</a>	75
7.15.2	<a href="#">Function Documentation</a>	75
7.15.2.1	<a href="#">e_get_acc(int *x, int *y, int *z)</a>	75
7.15.2.2	<a href="#">e_init_acc(void)</a>	76
7.16	<a href="#">a_d/e_accelerometer.h File Reference</a>	76
7.16.1	<a href="#">Detailed Description</a>	76
7.16.2	<a href="#">Function Documentation</a>	77
7.16.2.1	<a href="#">e_get_acc(int *x, int *y, int *z)</a>	77
7.16.2.2	<a href="#">e_init_acc(void)</a>	77
7.17	<a href="#">a_d/e_prox_timer2.c File Reference</a>	77
7.17.1	<a href="#">Detailed Description</a>	78
7.17.2	<a href="#">Function Documentation</a>	78
7.17.2.1	<a href="#">__attribute__((interrupt, auto_psv, shadow))</a>	78
7.17.2.2	<a href="#">e_get_ambient_light(unsigned int sensor_number)</a>	78
7.17.2.3	<a href="#">e_get_prox(unsigned int sensor_number)</a>	78
7.17.2.4	<a href="#">e_init_prox(void)</a>	79
7.17.2.5	<a href="#">e_stop_prox(void)</a>	79
7.17.2.6	<a href="#">init_tmr2(void)</a>	79
7.17.3	<a href="#">Variable Documentation</a>	79
7.17.3.1	<a href="#">ambient_and_reflected_ir</a>	79
7.17.3.2	<a href="#">ambient_ir</a>	79
7.17.3.3	<a href="#">reflected_ir</a>	79
7.18	<a href="#">acc_gyro/e_lsm330.c File Reference</a>	79
7.18.1	<a href="#">Detailed Description</a>	80
7.18.2	<a href="#">Macro Definition Documentation</a>	81

7.18.2.1	ACC_ADDR	81
7.18.2.2	GYRO_ADDR	81
7.18.3	Function Documentation	81
7.18.3.1	calibrateGyroscope(int numSamples)	81
7.18.3.2	getAllAxesAcc(signed int *x, signed int *y, signed int *z)	81
7.18.3.3	getAllAxesAccRaw(unsigned char *arr)	81
7.18.3.4	getAllAxesGyro(signed int *x, signed int *y, signed int *z)	81
7.18.3.5	getAllAxesGyroRaw(unsigned char *arr)	81
7.18.3.6	getTemperature(void)	81
7.18.3.7	getXAxisAcc()	81
7.18.3.8	getXAxisGyro()	81
7.18.3.9	getYAxisAcc()	81
7.18.3.10	getYAxisGyro()	81
7.18.3.11	getZAxisAcc()	81
7.18.3.12	getZAxisGyro()	81
7.18.3.13	initAccAndGyro(void)	81
7.18.3.14	rawToDpms(int value)	81
7.18.3.15	rawToDps(int value)	81
7.18.3.16	readReg(unsigned char addr, unsigned char reg)	81
7.18.3.17	readRegMulti(char device_add, unsigned char *read_buffer, char start_address, unsigned char numBytes)	82
7.18.3.18	writeReg(unsigned char addr, unsigned char reg, unsigned char value)	82
7.18.4	Variable Documentation	82
7.18.4.1	gyroOffsetX	82
7.18.4.2	gyroOffsetY	82
7.18.4.3	gyroOffsetZ	82
7.19	acc_gyro/e_lsm330.h File Reference	82
7.19.1	Detailed Description	83
7.19.2	Function Documentation	83
7.19.2.1	calibrateGyroscope()	83
7.19.2.2	getAllAxesAcc(signed int *x, signed int *y, signed int *z)	83

---

7.19.2.3	getAllAxesAccRaw(unsigned char *arr)	83
7.19.2.4	getAllAxesGyro(signed int *x, signed int *y, signed int *z)	83
7.19.2.5	getAllAxesGyroRaw(unsigned char *arr)	83
7.19.2.6	getTemperature(void)	83
7.19.2.7	getXAxisAcc()	83
7.19.2.8	getXAxisGyro()	83
7.19.2.9	getYAxisAcc()	84
7.19.2.10	getYAxisGyro()	84
7.19.2.11	getZAxisAcc()	84
7.19.2.12	getZAxisGyro()	84
7.19.2.13	initAccAndGyro(void)	84
7.19.2.14	rawToDpms(int value)	84
7.19.2.15	rawToDps(int value)	84
7.20	bluetooth/e_bluetooth.c File Reference	84
7.20.1	Detailed Description	85
7.20.2	Function Documentation	85
7.20.2.1	e_bt_connect_epuck(void)	85
7.20.2.2	e_bt_establish_SPP_link(char *address)	85
7.20.2.3	e_bt_exit_tranparent_mode(void)	86
7.20.2.4	e_bt_factory_reset(void)	86
7.20.2.5	e_bt_find_epuck(void)	86
7.20.2.6	e_bt_get_event_filter(void)	86
7.20.2.7	e_bt_get_friendly_name(struct BtDevice *device)	86
7.20.2.8	e_bt_inquiry(struct BtDevice *device)	87
7.20.2.9	e_bt_list_local_paired_device(void)	87
7.20.2.10	e_bt_read_local_name(char *name)	87
7.20.2.11	e_bt_read_local_pin_number(char *PIN)	88
7.20.2.12	e_bt_release_SPP_link(void)	88
7.20.2.13	e_bt_remove_local_paired_device(int j)	88
7.20.2.14	e_bt_reset(void)	88

---

7.20.2.15	<code>e_bt_send_SPP_data(char *data, char datalength)</code>	89
7.20.2.16	<code>e_bt_set_event_filter(char event)</code>	89
7.20.2.17	<code>e_bt_tranparent_mode(void)</code>	89
7.20.2.18	<code>e_bt_write_local_name(char *name)</code>	90
7.20.2.19	<code>e_bt_write_local_pin_number(char *PIN)</code>	91
7.20.3	Variable Documentation	91
7.20.3.1	<code>e_bt_local_paired_device</code>	91
7.20.3.2	<code>e_bt_present_device</code>	91
7.20.3.3	<code>e_bt_present_epuck</code>	91
7.20.3.4	<code>local_bt_PIN</code>	91
7.21	bluetooth/e_bluetooth.h File Reference	91
7.21.1	Detailed Description	93
7.21.2	Function Documentation	93
7.21.2.1	<code>e_bt_connect_epuck(void)</code>	93
7.21.2.2	<code>e_bt_establish_SPP_link(char *address)</code>	93
7.21.2.3	<code>e_bt_exit_tranparent_mode(void)</code>	93
7.21.2.4	<code>e_bt_factory_reset(void)</code>	93
7.21.2.5	<code>e_bt_find_epuck(void)</code>	94
7.21.2.6	<code>e_bt_get_event_filter(void)</code>	94
7.21.2.7	<code>e_bt_get_friendly_name(struct BtDevice *device)</code>	94
7.21.2.8	<code>e_bt_inquiry(struct BtDevice *device)</code>	94
7.21.2.9	<code>e_bt_list_local_paired_device(void)</code>	95
7.21.2.10	<code>e_bt_read_local_name(char *name)</code>	95
7.21.2.11	<code>e_bt_read_local_pin_number(char *PIN)</code>	95
7.21.2.12	<code>e_bt_release_SPP_link(void)</code>	96
7.21.2.13	<code>e_bt_remove_local_paired_device(int)</code>	96
7.21.2.14	<code>e_bt_reset(void)</code>	96
7.21.2.15	<code>e_bt_send_SPP_data(char *data, char datalenght)</code>	96
7.21.2.16	<code>e_bt_set_event_filter(char event)</code>	97
7.21.2.17	<code>e_bt_tranparent_mode(void)</code>	97



7.21.2.18	<code>e_bt_write_local_name(char *name)</code>	97
7.21.2.19	<code>e_bt_write_local_pin_number(char *PIN)</code>	97
7.21.3	Variable Documentation	98
7.21.3.1	<code>e_bt_local_paired_device</code>	98
7.21.3.2	<code>e_bt_present_device</code>	98
7.21.3.3	<code>e_bt_present_epuck</code>	98
7.22	<code>camera/fast_2_timer/e_calc_po3030k.c</code> File Reference	98
7.22.1	Detailed Description	99
7.22.2	Macro Definition Documentation	99
7.22.2.1	<code>ARRAY_ORIGINE_X</code>	99
7.22.2.2	<code>ARRAY_ORIGINE_Y</code>	99
7.22.2.3	<code>IRQ_PIX_LAT</code>	99
7.22.3	Function Documentation	99
7.22.3.1	<code>e_po3030k_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_factor_width, unsigned int zoom_factor_height, int color_mode)</code>	99
7.22.3.2	<code>e_po3030k_get_bytes_per_pixel(int color_mode)</code>	100
7.23	<code>camera/fast_2_timer/e_calc_po6030k.c</code> File Reference	100
7.23.1	Detailed Description	100
7.23.2	Macro Definition Documentation	101
7.23.2.1	<code>ARRAY_ORIGINE_X</code>	101
7.23.2.2	<code>ARRAY_ORIGINE_Y</code>	101
7.23.2.3	<code>IRQ_PIX_LAT</code>	101
7.23.3	Function Documentation	101
7.23.3.1	<code>e_po6030k_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_factor_width, unsigned int zoom_factor_height, int color_mode)</code>	101
7.23.3.2	<code>e_po6030k_get_bytes_per_pixel(int color_mode)</code>	101
7.24	<code>camera/fast_2_timer/e_calc_po8030d.c</code> File Reference	102
7.24.1	Detailed Description	102
7.24.2	Macro Definition Documentation	102
7.24.2.1	<code>ARRAY_ORIGINE_X</code>	102

7.24.2.2	ARRAY_ORIGINE_Y	102
7.24.2.3	IRQ_PIX_LAT	102
7.24.3	Function Documentation	102
7.24.3.1	e_po8030d_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact↔_width, unsigned int zoom_fact_height, int color_mode)	102
7.24.3.2	e_po8030d_get_bytes_per_pixel(int color_mode)	103
7.25	camera/fast_2_timer/e_common.c File Reference	103
7.25.1	Macro Definition Documentation	104
7.25.1.1	DEVICE_ID	104
7.25.2	Function Documentation	104
7.25.2.1	e_poxxxx_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact_width, unsigned int zoom_fact_height, int color_mode)	104
7.25.2.2	e_poxxxx_get_orientation(void)	104
7.25.2.3	e_poxxxx_init_cam(void)	104
7.25.2.4	e_poxxxx_set_awb_ae(int awb, int ae)	104
7.25.2.5	e_poxxxx_set_exposure(unsigned long exp)	105
7.25.2.6	e_poxxxx_set_mirror(int vertical, int horizontal)	105
7.25.2.7	e_poxxxx_set_rgb_gain(unsigned char r, unsigned char g, unsigned char b)	105
7.25.2.8	e_poxxxx_write_cam_registers(void)	105
7.25.2.9	getCameraVersion()	105
7.25.2.10	readee(unsigned page, unsigned addr)	105
7.25.3	Variable Documentation	105
7.25.3.1	camera_version	105
7.26	camera/fast_2_timer/e_po3030k.h File Reference	105
7.26.1	Detailed Description	107
7.26.2	Macro Definition Documentation	107
7.26.2.1	MODE_QQVGA	107
7.26.2.2	MODE_QVGA	107
7.26.2.3	MODE_VGA	107
7.26.2.4	PO3030K_FULL	107

7.26.2.5	SPEED_128	107
7.26.2.6	SPEED_16	107
7.26.2.7	SPEED_2	107
7.26.2.8	SPEED_2_3	107
7.26.2.9	SPEED_32	107
7.26.2.10	SPEED_4	107
7.26.2.11	SPEED_64	107
7.26.2.12	SPEED_8	107
7.26.3	Function Documentation	107
7.26.3.1	e_po3030k_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_factor, unsigned int zoom_factor_height, int color_mode)	107
7.26.3.2	e_po3030k_get_bytes_per_pixel(int color_mode)	108
7.26.3.3	e_po3030k_get_register(unsigned char adr, unsigned char *value)	109
7.26.3.4	e_po3030k_init_cam(void)	109
7.26.3.5	e_po3030k_read_cam_registers(void)	109
7.26.3.6	e_po3030k_set_adc_offset(unsigned char offset)	109
7.26.3.7	e_po3030k_set_ae_speed(unsigned char b, unsigned char d)	110
7.26.3.8	e_po3030k_set_awb_ae(int awb, int ae)	110
7.26.3.9	e_po3030k_set_awb_ae_tol(unsigned char awbm, unsigned char aem)	110
7.26.3.10	e_po3030k_set_bias(unsigned char pixbias, unsigned char opbias)	111
7.26.3.11	e_po3030k_set_brigh_contr(unsigned char bright, unsigned char contrast)	111
7.26.3.12	e_po3030k_set_cb_cr_gain(unsigned char cg11c, unsigned char cg22c)	111
7.26.3.13	e_po3030k_set_color_gain(unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)	112
7.26.3.14	e_po3030k_set_color_matrix(unsigned char array[3 * 3])	112
7.26.3.15	e_po3030k_set_color_mode(int mode)	112
7.26.3.16	e_po3030k_set_edge_prop(unsigned char gain, unsigned char tresh)	112
7.26.3.17	e_po3030k_set_exposure(long t)	113
7.26.3.18	e_po3030k_set_flicker_detection(int hz50, int hz60)	113
7.26.3.19	e_po3030k_set_flicker_man_set(int hz50, int hz60, int fdm, int fk, int tol)	113
7.26.3.20	e_po3030k_set_flicker_mode(int manual)	114

7.26.3.21	<code>e_po3030k_set_gamma_coef(unsigned char array[12], char color)</code> . . . . .	114
7.26.3.22	<code>e_po3030k_set_integr_time(unsigned long time)</code> . . . . .	115
7.26.3.23	<code>e_po3030k_set_lens_gain(unsigned char red, unsigned char green, unsigned char blue)</code> . . . . .	115
7.26.3.24	<code>e_po3030k_set_max_min_awb(unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob)</code> . . . . .	115
7.26.3.25	<code>e_po3030k_set_max_min_exp(unsigned int min, unsigned int max)</code> . . . . .	116
7.26.3.26	<code>e_po3030k_set_mirror(int vertical, int horizontal)</code> . . . . .	116
7.26.3.27	<code>e_po3030k_set_ref_exposure(unsigned char exp)</code> . . . . .	116
7.26.3.28	<code>e_po3030k_set_register(unsigned char adr, unsigned char value)</code> . . . . .	116
7.26.3.29	<code>e_po3030k_set_sampling_mode(int mode)</code> . . . . .	117
7.26.3.30	<code>e_po3030k_set_sepia(int status)</code> . . . . .	117
7.26.3.31	<code>e_po3030k_set_sepia_tone(unsigned char cb, unsigned char cr)</code> . . . . .	118
7.26.3.32	<code>e_po3030k_set_speed(int mode)</code> . . . . .	118
7.26.3.33	<code>e_po3030k_set_vsync(unsigned int start, unsigned int stop, unsigned int col)</code> . . . . .	118
7.26.3.34	<code>e_po3030k_set_weight_win(unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2)</code> . . . . .	119
7.26.3.35	<code>e_po3030k_set_ww(unsigned char ww)</code> . . . . .	119
7.26.3.36	<code>e_po3030k_set_wx(unsigned int start, unsigned int stop)</code> . . . . .	120
7.26.3.37	<code>e_po3030k_set_wy(unsigned int start, unsigned int stop)</code> . . . . .	120
7.26.3.38	<code>e_po3030k_sync_register_array(unsigned char start, unsigned char stop)</code> . . . . .	120
7.26.3.39	<code>e_po3030k_write_cam_registers(void)</code> . . . . .	122
7.26.3.40	<code>e_po3030k_write_gamma_coef(void)</code> . . . . .	122
7.27	<code>camera/slow_3_timer/e_po3030k.h</code> File Reference . . . . .	122
7.27.1	Detailed Description . . . . .	124
7.27.2	Macro Definition Documentation . . . . .	124
7.27.2.1	<code>ARRAY_HEIGHT</code> . . . . .	124
7.27.2.2	<code>ARRAY_WIDTH</code> . . . . .	124
7.27.2.3	<code>GREY_SCALE_MODE</code> . . . . .	124
7.27.2.4	<code>MODE_QQVGA</code> . . . . .	124
7.27.2.5	<code>MODE_QVGA</code> . . . . .	124
7.27.2.6	<code>MODE_VGA</code> . . . . .	124

7.27.2.7	PO3030K_FULL	124
7.27.2.8	RGB_565_MODE	125
7.27.2.9	SPEED_128	125
7.27.2.10	SPEED_16	125
7.27.2.11	SPEED_2	125
7.27.2.12	SPEED_2_3	125
7.27.2.13	SPEED_32	125
7.27.2.14	SPEED_4	125
7.27.2.15	SPEED_64	125
7.27.2.16	SPEED_8	125
7.27.2.17	YUV_MODE	125
7.27.3	Function Documentation	125
7.27.3.1	e_po3030k_apply_timer_config(int pixel_row, int pixel_col, int bpp, int pbp, int bbl)	125
7.27.3.2	e_po3030k_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact← _width, unsigned int zoom_fact_height, int color_mode)	125
7.27.3.3	e_po3030k_get_bytes_per_pixel(int color_mode)	127
7.27.3.4	e_po3030k_get_register(unsigned char adr, unsigned char *value)	127
7.27.3.5	e_po3030k_init_cam(void)	127
7.27.3.6	e_po3030k_is_img_ready(void)	127
7.27.3.7	e_po3030k_launch_capture(char *buf)	127
7.27.3.8	e_po3030k_read_cam_registers(void)	128
7.27.3.9	e_po3030k_set_adc_offset(unsigned char offset)	128
7.27.3.10	e_po3030k_set_ae_speed(unsigned char b, unsigned char d)	128
7.27.3.11	e_po3030k_set_awb_ae(int awb, int ae)	128
7.27.3.12	e_po3030k_set_awb_ae_tol(unsigned char awbm, unsigned char aem)	129
7.27.3.13	e_po3030k_set_bias(unsigned char pixbias, unsigned char opbias)	129
7.27.3.14	e_po3030k_set_brigh_contr(unsigned char bright, unsigned char contrast)	129
7.27.3.15	e_po3030k_set_cb_cr_gain(unsigned char cg11c, unsigned char cg22c)	130
7.27.3.16	e_po3030k_set_color_gain(unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)	130
7.27.3.17	e_po3030k_set_color_matrix(unsigned char array[3 *3])	130

7.27.3.18 e_po3030k_set_color_mode(int mode) . . . . .	130
7.27.3.19 e_po3030k_set_edge_prop(unsigned char gain, unsigned char tresh) . . . . .	131
7.27.3.20 e_po3030k_set_exposure(long t) . . . . .	131
7.27.3.21 e_po3030k_set_flicker_detection(int hz50, int hz60) . . . . .	131
7.27.3.22 e_po3030k_set_flicker_man_set(int hz50, int hz60, int fdm, int fk, int tol) . . . . .	132
7.27.3.23 e_po3030k_set_flicker_mode(int manual) . . . . .	132
7.27.3.24 e_po3030k_set_gamma_coef(unsigned char array[12], char color) . . . . .	133
7.27.3.25 e_po3030k_set_integr_time(unsigned long time) . . . . .	133
7.27.3.26 e_po3030k_set_lens_gain(unsigned char red, unsigned char green, unsigned char blue) . . . . .	133
7.27.3.27 e_po3030k_set_max_min_awb(unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob) . . . . .	134
7.27.3.28 e_po3030k_set_max_min_exp(unsigned int min, unsigned int max) . . . . .	134
7.27.3.29 e_po3030k_set_mirror(int vertical, int horizontal) . . . . .	134
7.27.3.30 e_po3030k_set_ref_exposure(unsigned char exp) . . . . .	135
7.27.3.31 e_po3030k_set_register(unsigned char adr, unsigned char value) . . . . .	135
7.27.3.32 e_po3030k_set_sampling_mode(int mode) . . . . .	135
7.27.3.33 e_po3030k_set_sepia(int status) . . . . .	136
7.27.3.34 e_po3030k_set_sepia_tone(unsigned char cb, unsigned char cr) . . . . .	136
7.27.3.35 e_po3030k_set_speed(int mode) . . . . .	136
7.27.3.36 e_po3030k_set_vsync(unsigned int start, unsigned int stop, unsigned int col) . . . . .	137
7.27.3.37 e_po3030k_set_weight_win(unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2) . . . . .	137
7.27.3.38 e_po3030k_set_ww(unsigned char ww) . . . . .	138
7.27.3.39 e_po3030k_set_wx(unsigned int start, unsigned int stop) . . . . .	138
7.27.3.40 e_po3030k_set_wy(unsigned int start, unsigned int stop) . . . . .	138
7.27.3.41 e_po3030k_sync_register_array(unsigned char start, unsigned char stop) . . . . .	139
7.27.3.42 e_po3030k_write_cam_registers(void) . . . . .	139
7.27.3.43 e_po3030k_write_gamma_coef(void) . . . . .	140
7.28 camera/fast_2_timer/e_po3030k_registers.c File Reference . . . . .	140
7.28.1 Detailed Description . . . . .	142
7.28.2 Macro Definition Documentation . . . . .	142

---

7.28.2.1	ADCOFF_BASE	142
7.28.2.2	AESPEED_BASE	142
7.28.2.3	ARRAY_ORIGINE_X	142
7.28.2.4	ARRAY_ORIGINE_Y	142
7.28.2.5	AWBAEENABLE_BASE	142
7.28.2.6	AWVAETOL_BASE	142
7.28.2.7	BASE_D1	142
7.28.2.8	BASE_D2	142
7.28.2.9	BASE_D3	142
7.28.2.10	BASE_D4	142
7.28.2.11	BIAS_BASE	142
7.28.2.12	BRICTR_BASE	143
7.28.2.13	CBCRGAIN_BASE	143
7.28.2.14	COLGAIN_BASE	143
7.28.2.15	COLOR_COEF_BASE	143
7.28.2.16	COLOR_M_ADDR	143
7.28.2.17	DEVICE_ID	143
7.28.2.18	EDGE_BASE	143
7.28.2.19	EXPOSURE_BASE	143
7.28.2.20	FLICKM_BASE	143
7.28.2.21	FLICKP_BASE	143
7.28.2.22	FRAME_HEIGHT	143
7.28.2.23	FRAME_WIDTH	143
7.28.2.24	GAMMA_BASE	143
7.28.2.25	GAMMASELCOL_BASE	143
7.28.2.26	INTEGR_BASE	143
7.28.2.27	LENSG_BASE	143
7.28.2.28	MCLK	143
7.28.2.29	MCLK_P_NS	143
7.28.2.30	MINMAXAWB_BASE	143

7.28.2.31 MINMAXEXP_BASE . . . . .	143
7.28.2.32 MIRROR_BASE . . . . .	143
7.28.2.33 MODE_GRAYSCALE . . . . .	143
7.28.2.34 MODE_R5G6B5 . . . . .	143
7.28.2.35 MODE_YUV . . . . .	144
7.28.2.36 NB_REGISTERS . . . . .	144
7.28.2.37 REFREPO_BASE . . . . .	144
7.28.2.38 SAMPLING_ADDR . . . . .	144
7.28.2.39 SEPIA_BASE . . . . .	144
7.28.2.40 SEPIATONE_BASE . . . . .	144
7.28.2.41 SPEED_ADDR . . . . .	144
7.28.2.42 VSYNC COL_BASE . . . . .	144
7.28.2.43 VSYNC START_BASE . . . . .	144
7.28.2.44 VSYNC STOP_BASE . . . . .	144
7.28.2.45 WEIGHWIN_BASE . . . . .	144
7.28.2.46 WINDOW_X1_BASE . . . . .	144
7.28.2.47 WINDOW_X2_BASE . . . . .	144
7.28.2.48 WINDOW_Y1_BASE . . . . .	144
7.28.2.49 WINDOW_Y2_BASE . . . . .	144
7.28.2.50 WW_BASE . . . . .	144
7.28.3 Function Documentation . . . . .	144
7.28.3.1 e_po3030k_get_register(unsigned char adr, unsigned char *value) . . . . .	144
7.28.3.2 e_po3030k_read_cam_registers(void) . . . . .	145
7.28.3.3 e_po3030k_set_adc_offset(unsigned char offset) . . . . .	145
7.28.3.4 e_po3030k_set_ae_speed(unsigned char b, unsigned char d) . . . . .	145
7.28.3.5 e_po3030k_set_awb_ae(int awb, int ae) . . . . .	146
7.28.3.6 e_po3030k_set_awb_ae_tol(unsigned char awbm, unsigned char aem) . . . . .	146
7.28.3.7 e_po3030k_set_bias(unsigned char pixbias, unsigned char opbias) . . . . .	146
7.28.3.8 e_po3030k_set_bright_contr(unsigned char bright, unsigned char contrast) . . . . .	146
7.28.3.9 e_po3030k_set_cb_cr_gain(unsigned char cg11c, unsigned char cg22c) . . . . .	147



7.28.3.10 e_po3030k_set_color_gain(unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue) . . . . .	147
7.28.3.11 e_po3030k_set_color_mode(int mode) . . . . .	148
7.28.3.12 e_po3030k_set_edge_prop(unsigned char gain, unsigned char tresh) . . . . .	148
7.28.3.13 e_po3030k_set_exposure(long t) . . . . .	148
7.28.3.14 e_po3030k_set_flicker_detection(int hz50, int hz60) . . . . .	149
7.28.3.15 e_po3030k_set_flicker_man_set(int hz50, int hz60, int fdm, int fk, int tol) . . . . .	149
7.28.3.16 e_po3030k_set_flicker_mode(int manual) . . . . .	149
7.28.3.17 e_po3030k_set_gamma_coef(unsigned char array[12], char color) . . . . .	150
7.28.3.18 e_po3030k_set_integr_time(unsigned long time) . . . . .	150
7.28.3.19 e_po3030k_set_lens_gain(unsigned char red, unsigned char green, unsigned char blue) . . . . .	150
7.28.3.20 e_po3030k_set_max_min_awb(unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob) . . . . .	151
7.28.3.21 e_po3030k_set_max_min_exp(unsigned int max, unsigned int min) . . . . .	151
7.28.3.22 e_po3030k_set_mirror(int vertical, int horizontal) . . . . .	151
7.28.3.23 e_po3030k_set_ref_exposure(unsigned char exp) . . . . .	152
7.28.3.24 e_po3030k_set_register(unsigned char adr, unsigned char value) . . . . .	152
7.28.3.25 e_po3030k_set_sampling_mode(int mode) . . . . .	152
7.28.3.26 e_po3030k_set_sepia(int status) . . . . .	153
7.28.3.27 e_po3030k_set_sepia_tone(unsigned char cb, unsigned char cr) . . . . .	153
7.28.3.28 e_po3030k_set_speed(int mode) . . . . .	153
7.28.3.29 e_po3030k_set_vsync(unsigned int start, unsigned int stop, unsigned int col) . . . . .	154
7.28.3.30 e_po3030k_set_weight_win(unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2) . . . . .	154
7.28.3.31 e_po3030k_set_ww(unsigned char ww) . . . . .	155
7.28.3.32 e_po3030k_set_wx(unsigned int start, unsigned int stop) . . . . .	155
7.28.3.33 e_po3030k_set_wy(unsigned int start, unsigned int stop) . . . . .	155
7.28.3.34 e_po3030k_SetColorMatrix(unsigned char array[3 *3]) . . . . .	156
7.28.3.35 e_po3030k_sync_register_array(unsigned char start, unsigned char stop) . . . . .	156
7.28.3.36 e_po3030k_write_cam_registers(void) . . . . .	157
7.28.3.37 e_po3030k_write_gamma_coef(void) . . . . .	157

7.28.3.38	<code>po3030k_get_pixelclock(void)</code> . . . . .	157
7.28.4	Variable Documentation . . . . .	157
7.28.4.1	<code>cam_reg</code> . . . . .	157
7.29	<code>camera/fast_2_timer/e_po6030k.h</code> File Reference . . . . .	157
7.29.1	Detailed Description . . . . .	158
7.29.2	Macro Definition Documentation . . . . .	158
7.29.2.1	<code>BANK_A</code> . . . . .	158
7.29.2.2	<code>BANK_B</code> . . . . .	158
7.29.2.3	<code>BANK_C</code> . . . . .	158
7.29.2.4	<code>BANK_D</code> . . . . .	158
7.29.2.5	<code>BAYER_CLOCK_1</code> . . . . .	158
7.29.2.6	<code>BAYER_CLOCK_2</code> . . . . .	158
7.29.2.7	<code>BAYER_CLOCK_4</code> . . . . .	158
7.29.2.8	<code>BAYER_CLOCK_8</code> . . . . .	158
7.29.2.9	<code>e_po6030k_set_speed</code> . . . . .	158
7.29.2.10	<code>E_PO6030K_SKETCH_BW</code> . . . . .	159
7.29.2.11	<code>E_PO6030K_SKETCH_COLOR</code> . . . . .	159
7.29.2.12	<code>PO_6030_MODE_QQVGA</code> . . . . .	159
7.29.2.13	<code>PO_6030_MODE_QVGA</code> . . . . .	159
7.29.2.14	<code>PO_6030_MODE_VGA</code> . . . . .	159
7.29.2.15	<code>PO_6030_SPEED_1</code> . . . . .	159
7.29.2.16	<code>PO_6030_SPEED_2</code> . . . . .	159
7.29.2.17	<code>PO_6030_SPEED_4</code> . . . . .	159
7.29.2.18	<code>PO_6030_SPEED_8</code> . . . . .	159
7.29.3	Function Documentation . . . . .	159
7.29.3.1	<code>e_po6030k_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_factor, unsigned int zoom_factor_height, int color_mode)</code> . . . . .	159
7.29.3.2	<code>e_po6030k_get_bytes_per_pixel(int color_mode)</code> . . . . .	160
7.29.3.3	<code>e_po6030k_set_awb_ae(int awb, int ae)</code> . . . . .	161
7.29.3.4	<code>e_po6030k_set_bank(unsigned char bank)</code> . . . . .	161

7.29.3.5	<a href="#">e_po6030k_set_bayer_clkdiv(unsigned char div)</a>	161
7.29.3.6	<a href="#">e_po6030k_set_color_mode(unsigned char format)</a>	161
7.29.3.7	<a href="#">e_po6030k_set_exposure(unsigned long exp)</a>	162
7.29.3.8	<a href="#">e_po6030k_set_mirror(int vertical, int horizontal)</a>	163
7.29.3.9	<a href="#">e_po6030k_set_mode(unsigned char format, unsigned char sampl_mode)</a>	163
7.29.3.10	<a href="#">e_po6030k_set_pclkdiv(unsigned char div)</a>	163
7.29.3.11	<a href="#">e_po6030k_set_rgb_gain(unsigned char r, unsigned char g, unsigned char b)</a>	163
7.29.3.12	<a href="#">e_po6030k_set_sketch_mode(int mode)</a>	163
7.29.3.13	<a href="#">e_po6030k_set_vsync(unsigned int start, unsigned int stop)</a>	164
7.29.3.14	<a href="#">e_po6030k_set_wx(unsigned int start, unsigned int stop)</a>	164
7.29.3.15	<a href="#">e_po6030k_set_wy(unsigned int start, unsigned int stop)</a>	164
7.29.3.16	<a href="#">e_po6030k_write_register(unsigned char bank, unsigned char reg, unsigned char value)</a>	165
7.30	<a href="#">camera/fast_2_timer/e_po6030k_registers.c File Reference</a>	165
7.30.1	<a href="#">Detailed Description</a>	166
7.30.2	<a href="#">Macro Definition Documentation</a>	166
7.30.2.1	<a href="#">ARRAY_ORIGINE_X</a>	166
7.30.2.2	<a href="#">ARRAY_ORIGINE_Y</a>	166
7.30.2.3	<a href="#">BANK_REGISTER</a>	166
7.30.2.4	<a href="#">DEVICE_ID</a>	166
7.30.2.5	<a href="#">MCLK</a>	166
7.30.2.6	<a href="#">MCLK_P_NS</a>	166
7.30.3	<a href="#">Function Documentation</a>	166
7.30.3.1	<a href="#">e_po6030k_read_register(unsigned char bank, unsigned char reg)</a>	166
7.30.3.2	<a href="#">e_po6030k_set_awb_ae(int awb, int ae)</a>	166
7.30.3.3	<a href="#">e_po6030k_set_bank(unsigned char bank)</a>	167
7.30.3.4	<a href="#">e_po6030k_set_bayer_clkdiv(unsigned char div)</a>	167
7.30.3.5	<a href="#">e_po6030k_set_exposure(unsigned long exp)</a>	167
7.30.3.6	<a href="#">e_po6030k_set_mirror(int vertical, int horizontal)</a>	167
7.30.3.7	<a href="#">e_po6030k_set_mode(unsigned char format, unsigned char sampl_mode)</a>	168
7.30.3.8	<a href="#">e_po6030k_set_pclkdiv(unsigned char div)</a>	168

7.30.3.9	<code>e_po6030k_set_rgb_gain(unsigned char r, unsigned char g, unsigned char b)</code>	168
7.30.3.10	<code>e_po6030k_set_sampl_color(unsigned char sample)</code>	168
7.30.3.11	<code>e_po6030k_set_sampl_gray(unsigned char sample)</code>	168
7.30.3.12	<code>e_po6030k_set_sketch_mode(int mode)</code>	168
7.30.3.13	<code>e_po6030k_set_vsync(unsigned int start, unsigned int stop)</code>	168
7.30.3.14	<code>e_po6030k_set_wx(unsigned int start, unsigned int stop)</code>	169
7.30.3.15	<code>e_po6030k_set_wy(unsigned int start, unsigned int stop)</code>	169
7.30.3.16	<code>e_po6030k_write_register(unsigned char bank, unsigned char reg, unsigned char value)</code>	169
7.31	camera/fast_2_timer/e_po8030d.h File Reference	170
7.31.1	Detailed Description	171
7.31.2	Macro Definition Documentation	171
7.31.2.1	<code>BANK_A</code>	171
7.31.2.2	<code>BANK_B</code>	171
7.31.2.3	<code>BANK_C</code>	171
7.31.2.4	<code>BANK_D</code>	171
7.31.2.5	<code>e_po8030d_set_speed</code>	171
7.31.2.6	<code>E_PO8030D_SKETCH_BW</code>	171
7.31.2.7	<code>E_PO8030D_SKETCH_COLOR</code>	171
7.31.2.8	<code>PO_8030_BAYER_CLOCK_1</code>	171
7.31.2.9	<code>PO_8030_BAYER_CLOCK_2</code>	171
7.31.2.10	<code>PO_8030_BAYER_CLOCK_4</code>	171
7.31.2.11	<code>PO_8030_BAYER_CLOCK_8</code>	171
7.31.2.12	<code>PO_8030_MODE_QQVGA</code>	171
7.31.2.13	<code>PO_8030_MODE_QVGA</code>	171
7.31.2.14	<code>PO_8030_MODE_VGA</code>	171
7.31.2.15	<code>PO_8030_SPEED_1</code>	171
7.31.2.16	<code>PO_8030_SPEED_2</code>	171
7.31.2.17	<code>PO_8030_SPEED_4</code>	171
7.31.2.18	<code>PO_8030_SPEED_8</code>	171
7.31.3	Function Documentation	171

7.31.3.1	<code>e_po8030d_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_factor, unsigned int zoom_factor_height, int color_mode)</code>	171
7.31.3.2	<code>e_po8030d_get_bytes_per_pixel(int color_mode)</code>	172
7.31.3.3	<code>e_po8030d_set_awb_ae(int awb, int ae)</code>	172
7.31.3.4	<code>e_po8030d_set_bank(unsigned char bank)</code>	172
7.31.3.5	<code>e_po8030d_set_bayer_clkdiv(unsigned char div)</code>	173
7.31.3.6	<code>e_po8030d_set_brightness(signed char value)</code>	173
7.31.3.7	<code>e_po8030d_set_color_mode(unsigned char format)</code>	173
7.31.3.8	<code>e_po8030d_set_exposure(unsigned long exp)</code>	173
7.31.3.9	<code>e_po8030d_set_mirror(int vertical, int horizontal)</code>	173
7.31.3.10	<code>e_po8030d_set_mode(unsigned char format, unsigned char sample_mode)</code>	174
7.31.3.11	<code>e_po8030d_set_pclkdiv(unsigned char div)</code>	174
7.31.3.12	<code>e_po8030d_set_rgb_gain(unsigned char r, unsigned char g, unsigned char b)</code>	174
7.31.3.13	<code>e_po8030d_set_sketch_mode(int mode)</code>	174
7.31.3.14	<code>e_po8030d_set_vsync(unsigned int start, unsigned int stop)</code>	174
7.31.3.15	<code>e_po8030d_set_wx(unsigned int start, unsigned int stop)</code>	175
7.31.3.16	<code>e_po8030d_set_wy(unsigned int start, unsigned int stop)</code>	175
7.31.3.17	<code>e_po8030d_write_register(unsigned char bank, unsigned char reg, unsigned char value)</code>	175
7.31.3.18	<code>init_po8030()</code>	176
7.31.3.19	<code>testWriteReg(unsigned char bank)</code>	176
7.32	<code>camera/fast_2_timer/e_po8030d_registers.c</code> File Reference	176
7.32.1	Detailed Description	177
7.32.2	Macro Definition Documentation	177
7.32.2.1	<code>ARRAY_ORIGINE_X</code>	177
7.32.2.2	<code>ARRAY_ORIGINE_Y</code>	177
7.32.2.3	<code>BANK_REGISTER</code>	177
7.32.2.4	<code>DEVICE_ID</code>	177
7.32.2.5	<code>MCLK</code>	177
7.32.2.6	<code>MCLK_P_NS</code>	177
7.32.3	Function Documentation	177

7.32.3.1	<code>e_po8030d_read_register(unsigned char bank, unsigned char reg)</code>	177
7.32.3.2	<code>e_po8030d_set_awb_ae(int awb, int ae)</code>	177
7.32.3.3	<code>e_po8030d_set_bank(unsigned char bank)</code>	178
7.32.3.4	<code>e_po8030d_set_bayer_clkdiv(unsigned char div)</code>	178
7.32.3.5	<code>e_po8030d_set_brightness(signed char value)</code>	178
7.32.3.6	<code>e_po8030d_set_exposure(unsigned long exp)</code>	178
7.32.3.7	<code>e_po8030d_set_mirror(int vertical, int horizontal)</code>	179
7.32.3.8	<code>e_po8030d_set_mode(unsigned char format, unsigned char sampl_mode)</code>	179
7.32.3.9	<code>e_po8030d_set_pclkdiv(unsigned char div)</code>	179
7.32.3.10	<code>e_po8030d_set_rgb_gain(unsigned char r, unsigned char g, unsigned char b)</code>	179
7.32.3.11	<code>e_po8030d_set_sampl_color(unsigned char sample)</code>	179
7.32.3.12	<code>e_po8030d_set_sampl_gray(unsigned char sample)</code>	179
7.32.3.13	<code>e_po8030d_set_sketch_mode(int mode)</code>	179
7.32.3.14	<code>e_po8030d_set_vsync(unsigned int start, unsigned int stop)</code>	180
7.32.3.15	<code>e_po8030d_set_wx(unsigned int start, unsigned int stop)</code>	180
7.32.3.16	<code>e_po8030d_set_wy(unsigned int start, unsigned int stop)</code>	180
7.32.3.17	<code>e_po8030d_write_register(unsigned char bank, unsigned char reg, unsigned char value)</code>	181
7.32.3.18	<code>enablePO8030()</code>	181
7.32.3.19	<code>init_po8030()</code>	181
7.32.3.20	<code>testWriteReg(unsigned char bank)</code>	181
7.33	<code>camera/fast_2_timer/e_poxxxx.h</code> File Reference	181
7.33.1	Detailed Description	182
7.33.2	Macro Definition Documentation	182
7.33.2.1	<code>ARRAY_HEIGHT</code>	182
7.33.2.2	<code>ARRAY_WIDTH</code>	182
7.33.2.3	<code>GREY_SCALE_MODE</code>	182
7.33.2.4	<code>POXXXX_FULL</code>	182
7.33.2.5	<code>RGB_565_MODE</code>	182
7.33.2.6	<code>YUV_MODE</code>	182
7.33.3	Function Documentation	182

7.33.3.1	<code>e_poxxxx_apply_timer_config(int pixel_row, int pixel_col, int bpp, int pbp, int bbl)</code>	182
7.33.3.2	<code>e_poxxxx_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact_width, unsigned int zoom_fact_height, int color_mode)</code>	183
7.33.3.3	<code>e_poxxxx_get_orientation(void)</code>	183
7.33.3.4	<code>e_poxxxx_init_cam(void)</code>	183
7.33.3.5	<code>e_poxxxx_is_img_ready(void)</code>	183
7.33.3.6	<code>e_poxxxx_launch_capture(char *buf)</code>	183
7.33.3.7	<code>e_poxxxx_set_awb_ae(int awb, int ae)</code>	184
7.33.3.8	<code>e_poxxxx_set_exposure(unsigned long exp)</code>	185
7.33.3.9	<code>e_poxxxx_set_mirror(int vertical, int horizontal)</code>	185
7.33.3.10	<code>e_poxxxx_set_rgb_gain(unsigned char r, unsigned char g, unsigned char b)</code>	185
7.33.3.11	<code>e_poxxxx_write_cam_registers(void)</code>	185
7.34	camera/fast_2_timer/e_timers.c File Reference	185
7.34.1	Detailed Description	186
7.34.2	Function Documentation	186
7.34.2.1	<code>__attribute__((near))</code>	186
7.34.2.2	<code>e_poxxxx_apply_timer_config(int pixel_row, int pixel_col, int bpp, int pbp, int bbl)</code>	186
7.34.2.3	<code>e_poxxxx_is_img_ready(void)</code>	187
7.34.2.4	<code>e_poxxxx_launch_capture(char *buf)</code>	187
7.34.2.5	<code>init_timer4(void)</code>	187
7.34.2.6	<code>init_timer5(void)</code>	187
7.34.3	Variable Documentation	187
7.34.3.1	<code>_poxxxx_buffer</code>	187
7.34.3.2	<code>_poxxxx_img_ready</code>	188
7.34.3.3	<code>blank_row_betw</code>	188
7.35	camera/slow_3_timer/e_timers.c File Reference	188
7.35.1	Detailed Description	189
7.35.2	Function Documentation	189
7.35.2.1	<code>__attribute__((interrupt, auto_psv))</code>	189
7.35.2.2	<code>e_po3030k_apply_timer_config(int pixel_row, int pixel_col, int bpp, int pbp, int bbl)</code>	189

7.35.2.3	<code>e_po3030k_is_img_ready(void)</code> . . . . .	189
7.35.2.4	<code>e_po3030k_launch_capture(char *buf)</code> . . . . .	190
7.35.2.5	<code>init_timer1(void)</code> . . . . .	190
7.35.2.6	<code>init_timer4(void)</code> . . . . .	190
7.35.2.7	<code>init_timer5(void)</code> . . . . .	190
7.35.3	Variable Documentation . . . . .	190
7.35.3.1	<code>bbl_current</code> . . . . .	190
7.35.3.2	<code>blank_betw_lines</code> . . . . .	190
7.35.3.3	<code>bpp_current</code> . . . . .	190
7.35.3.4	<code>buf_pos</code> . . . . .	190
7.35.3.5	<code>buffer</code> . . . . .	190
7.35.3.6	<code>bytes_per_pixel</code> . . . . .	190
7.35.3.7	<code>current_col</code> . . . . .	191
7.35.3.8	<code>current_row</code> . . . . .	191
7.35.3.9	<code>img_ready</code> . . . . .	191
7.35.3.10	<code>max_col</code> . . . . .	191
7.35.3.11	<code>max_row</code> . . . . .	191
7.35.3.12	<code>pbp_current</code> . . . . .	191
7.35.3.13	<code>pixel_betw_pixel</code> . . . . .	191
7.36	<code>camera/slow_3_timer/e_calc.c</code> File Reference . . . . .	192
7.36.1	Detailed Description . . . . .	192
7.36.2	Macro Definition Documentation . . . . .	192
7.36.2.1	<code>ARRAY_ORIGINE_X</code> . . . . .	192
7.36.2.2	<code>ARRAY_ORIGINE_Y</code> . . . . .	192
7.36.3	Function Documentation . . . . .	192
7.36.3.1	<code>e_po3030k_config_cam(unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_factor, unsigned int zoom_factor_height, int color_mode)</code> . . . . .	192
7.36.3.2	<code>e_po3030k_get_bytes_per_pixel(int color_mode)</code> . . . . .	193
7.36.3.3	<code>e_po3030k_init_cam(void)</code> . . . . .	193
7.37	<code>camera/slow_3_timer/e_registers.c</code> File Reference . . . . .	193



7.37.1 Detailed Description . . . . .	195
7.37.2 Macro Definition Documentation . . . . .	196
7.37.2.1 ADCOFF_BASE . . . . .	196
7.37.2.2 AESPEED_BASE . . . . .	196
7.37.2.3 ARRAY_ORIGINE_X . . . . .	196
7.37.2.4 ARRAY_ORIGINE_Y . . . . .	196
7.37.2.5 AWBAEENABLE_BASE . . . . .	196
7.37.2.6 AWVAETOL_BASE . . . . .	196
7.37.2.7 BASE_D1 . . . . .	196
7.37.2.8 BASE_D2 . . . . .	196
7.37.2.9 BASE_D3 . . . . .	196
7.37.2.10 BASE_D4 . . . . .	196
7.37.2.11 BIAS_BASE . . . . .	196
7.37.2.12 BRICTR_BASE . . . . .	196
7.37.2.13 CBCRGAIN_BASE . . . . .	196
7.37.2.14 COLGAIN_BASE . . . . .	196
7.37.2.15 COLOR_COEF_BASE . . . . .	196
7.37.2.16 COLOR_M_ADDR . . . . .	196
7.37.2.17 DEVICE_ID . . . . .	196
7.37.2.18 EDGE_BASE . . . . .	196
7.37.2.19 EXPOSURE_BASE . . . . .	196
7.37.2.20 FLICKM_BASE . . . . .	196
7.37.2.21 FLICKP_BASE . . . . .	196
7.37.2.22 FRAME_HEIGTH . . . . .	196
7.37.2.23 FRAME_WIDTH . . . . .	197
7.37.2.24 GAMMA_BASE . . . . .	197
7.37.2.25 GAMMASELCOL_BASE . . . . .	197
7.37.2.26 INTEGR_BASE . . . . .	197
7.37.2.27 LENS_G_BASE . . . . .	197
7.37.2.28 MCLK . . . . .	197

7.37.2.29 MCLK_P_NS . . . . .	197
7.37.2.30 MINMAXAWB_BASE . . . . .	197
7.37.2.31 MINMAXEXP_BASE . . . . .	197
7.37.2.32 MIRROR_BASE . . . . .	197
7.37.2.33 MODE_GRAYSCALE . . . . .	197
7.37.2.34 MODE_R5G6B5 . . . . .	197
7.37.2.35 MODE_YUV . . . . .	197
7.37.2.36 NB_REGISTERS . . . . .	197
7.37.2.37 REFREPO_BASE . . . . .	197
7.37.2.38 SAMPLING_ADDR . . . . .	197
7.37.2.39 SEPIA_BASE . . . . .	197
7.37.2.40 SEPIATONE_BASE . . . . .	197
7.37.2.41 SPEED_ADDR . . . . .	197
7.37.2.42 VSYNCCOL_BASE . . . . .	197
7.37.2.43 VSYNCSTART_BASE . . . . .	197
7.37.2.44 VSYNCSTOP_BASE . . . . .	197
7.37.2.45 WEIGHWIN_BASE . . . . .	197
7.37.2.46 WINDOW_X1_BASE . . . . .	198
7.37.2.47 WINDOW_X2_BASE . . . . .	198
7.37.2.48 WINDOW_Y1_BASE . . . . .	198
7.37.2.49 WINDOW_Y2_BASE . . . . .	198
7.37.2.50 WW_BASE . . . . .	198
7.37.3 Function Documentation . . . . .	198
7.37.3.1 e_po3030k_get_register(unsigned char adr, unsigned char *value) . . . . .	198
7.37.3.2 e_po3030k_read_cam_registers(void) . . . . .	198
7.37.3.3 e_po3030k_set_adc_offset(unsigned char offset) . . . . .	198
7.37.3.4 e_po3030k_set_ae_speed(unsigned char b, unsigned char d) . . . . .	199
7.37.3.5 e_po3030k_set_awb_ae(int awb, int ae) . . . . .	199
7.37.3.6 e_po3030k_set_awb_ae_tol(unsigned char awbm, unsigned char aem) . . . . .	199
7.37.3.7 e_po3030k_set_bias(unsigned char pixbias, unsigned char opbias) . . . . .	200

7.37.3.8	<code>e_po3030k_set_brigh_contr(unsigned char bright, unsigned char contrast)</code>	200
7.37.3.9	<code>e_po3030k_set_cb_cr_gain(unsigned char cg11c, unsigned char cg22c)</code>	200
7.37.3.10	<code>e_po3030k_set_color_gain(unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)</code>	201
7.37.3.11	<code>e_po3030k_set_color_mode(int mode)</code>	201
7.37.3.12	<code>e_po3030k_set_edge_prop(unsigned char gain, unsigned char tresh)</code>	201
7.37.3.13	<code>e_po3030k_set_exposure(long t)</code>	202
7.37.3.14	<code>e_po3030k_set_flicker_detection(int hz50, int hz60)</code>	202
7.37.3.15	<code>e_po3030k_set_flicker_man_set(int hz50, int hz60, int fdm, int fk, int tol)</code>	202
7.37.3.16	<code>e_po3030k_set_flicker_mode(int manual)</code>	203
7.37.3.17	<code>e_po3030k_set_gamma_coef(unsigned char array[12], char color)</code>	203
7.37.3.18	<code>e_po3030k_set_integr_time(unsigned long time)</code>	204
7.37.3.19	<code>e_po3030k_set_lens_gain(unsigned char red, unsigned char green, unsigned char blue)</code>	204
7.37.3.20	<code>e_po3030k_set_max_min_awb(unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob)</code>	204
7.37.3.21	<code>e_po3030k_set_max_min_exp(unsigned int max, unsigned int min)</code>	205
7.37.3.22	<code>e_po3030k_set_mirror(int vertical, int horizontal)</code>	205
7.37.3.23	<code>e_po3030k_set_ref_exposure(unsigned char exp)</code>	205
7.37.3.24	<code>e_po3030k_set_register(unsigned char adr, unsigned char value)</code>	205
7.37.3.25	<code>e_po3030k_set_sampling_mode(int mode)</code>	206
7.37.3.26	<code>e_po3030k_set_sepia(int status)</code>	206
7.37.3.27	<code>e_po3030k_set_sepia_tone(unsigned char cb, unsigned char cr)</code>	207
7.37.3.28	<code>e_po3030k_set_speed(int mode)</code>	207
7.37.3.29	<code>e_po3030k_set_vsync(unsigned int start, unsigned int stop, unsigned int col)</code>	207
7.37.3.30	<code>e_po3030k_set_weight_win(unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2)</code>	208
7.37.3.31	<code>e_po3030k_set_ww(unsigned char ww)</code>	208
7.37.3.32	<code>e_po3030k_set_wx(unsigned int start, unsigned int stop)</code>	209
7.37.3.33	<code>e_po3030k_set_wy(unsigned int start, unsigned int stop)</code>	209
7.37.3.34	<code>e_po3030k_SetColorMatrix(unsigned char array[3 * 3])</code>	209
7.37.3.35	<code>e_po3030k_sync_register_array(unsigned char start, unsigned char stop)</code>	210

7.37.3.36	<code>e_po3030k_write_cam_registers(void)</code>	210
7.37.3.37	<code>e_po3030k_write_gamma_coef(void)</code>	210
7.37.3.38	<code>po3030k_get_pixclock(void)</code>	211
7.37.4	Variable Documentation	211
7.37.4.1	<code>cam_reg</code>	211
7.38	<code>codec/e_common.inc</code> File Reference	211
7.39	<code>codec/e_sound.c</code> File Reference	211
7.39.1	Detailed Description	211
7.39.2	Function Documentation	211
7.39.2.1	<code>e_close_sound(void)</code>	211
7.39.2.2	<code>e_init_sound(void)</code>	211
7.39.2.3	<code>e_play_sound(int sound_nbr, int sound_length)</code>	211
7.40	<code>codec/e_sound.h</code> File Reference	212
7.40.1	Detailed Description	212
7.40.2	Function Documentation	212
7.40.2.1	<code>e_close_sound(void)</code>	212
7.40.2.2	<code>e_init_codec_slave(void)</code>	213
7.40.2.3	<code>e_init_dci_master(void)</code>	213
7.40.2.4	<code>e_init_sound(void)</code>	213
7.40.2.5	<code>e_play_sound(int sound_offset, int sound_length)</code>	213
7.40.2.6	<code>e_sub_dci_kickoff(int, int)</code>	214
7.41	<code>contrib/LIS_sensors_turret/e_devantech.c</code> File Reference	214
7.41.1	Function Documentation	214
7.41.1.1	<code>e_disable_devantech(void)</code>	214
7.41.1.2	<code>e_get_delay_devantech(char device_add)</code>	214
7.41.1.3	<code>e_get_dist_devantech(char device_add)</code>	214
7.41.1.4	<code>e_get_light_devantech(char device_add)</code>	214
7.41.1.5	<code>e_get_sr_devantech(char device_add)</code>	214
7.41.1.6	<code>e_i2cd_readb(char device_add, char reg)</code>	214
7.41.1.7	<code>e_i2cd_write(char device_add, char reg, char value)</code>	215

7.41.1.8	<code>e_init_devantech(void)</code> . . . . .	215
7.41.1.9	<code>e_set_gain_devantech(char device_add, char gain)</code> . . . . .	215
7.41.1.10	<code>e_set_range_devantech(char device_add, char range)</code> . . . . .	215
7.42	<code>contrib/LIS_sensors_turret/e_devantech.h</code> File Reference . . . . .	215
7.42.1	Detailed Description . . . . .	215
7.42.2	Function Documentation . . . . .	216
7.42.2.1	<code>e_disable_devantech(void)</code> . . . . .	216
7.42.2.2	<code>e_get_delay_devantech(char device_add)</code> . . . . .	216
7.42.2.3	<code>e_get_dist_devantech(char device_add)</code> . . . . .	216
7.42.2.4	<code>e_get_light_devantech(char device_add)</code> . . . . .	216
7.42.2.5	<code>e_get_sr_devantech(char device_add)</code> . . . . .	216
7.42.2.6	<code>e_i2cd_readb(char device_add, char reg)</code> . . . . .	216
7.42.2.7	<code>e_i2cd_readw(char device_add, char reg)</code> . . . . .	216
7.42.2.8	<code>e_i2cd_write(char device_add, char reg, char value)</code> . . . . .	216
7.42.2.9	<code>e_init_devantech(void)</code> . . . . .	216
7.42.2.10	<code>e_set_gain_devantech(char device_add, char gain)</code> . . . . .	216
7.42.2.11	<code>e_set_range_devantech(char device_add, char range)</code> . . . . .	216
7.43	<code>contrib/LIS_sensors_turret/e_sensex.c</code> File Reference . . . . .	216
7.43.1	Function Documentation . . . . .	217
7.43.1.1	<code>e_get_sensex_wait(void)</code> . . . . .	217
7.43.1.2	<code>e_init_sensex(void)</code> . . . . .	217
7.43.1.3	<code>e_sensex_process(int *sensex_param, unsigned int *sensex_value)</code> . . . . .	217
7.43.1.4	<code>e_start_sensex_wait(void)</code> . . . . .	217
7.43.1.5	<code>e_stop_sensex_wait(void)</code> . . . . .	217
7.43.2	Variable Documentation . . . . .	217
7.43.2.1	<code>sensex_wait</code> . . . . .	217
7.44	<code>contrib/LIS_sensors_turret/e_sensex.h</code> File Reference . . . . .	217
7.44.1	Macro Definition Documentation . . . . .	218
7.44.1.1	<code>I2C_ADDR_CMPS03</code> . . . . .	218
7.44.1.2	<code>I2C_ADDR_SENSEXT</code> . . . . .	218

7.44.1.3	I2C_ADDR_SRF08	218
7.44.1.4	I2C_ADDR_SRF10	218
7.44.1.5	I2C_ADDR_SRF235	218
7.44.2	Function Documentation	218
7.44.2.1	e_get_sensex_wait(void)	218
7.44.2.2	e_init_sensex(void)	218
7.44.2.3	e_sensex_process(int *sensex_param, unsigned int *sensex_value)	218
7.44.2.4	e_start_sensex_wait(void)	218
7.44.2.5	e_stop_sensex_wait(void)	218
7.45	contrib/LIS_sensors_turret/e_sharp.c File Reference	218
7.45.1	Function Documentation	219
7.45.1.1	e_get_dist_sharp()	219
7.45.1.2	e_init_sharp(void)	219
7.45.1.3	e_set_sharp_led(unsigned int sharp_led_number, unsigned int value)	219
7.45.1.4	e_sharp_led_clear(void)	219
7.45.1.5	e_sharp_off(void)	219
7.45.1.6	e_sharp_on(void)	219
7.46	contrib/LIS_sensors_turret/e_sharp.h File Reference	219
7.46.1	Macro Definition Documentation	220
7.46.1.1	SHARP	220
7.46.1.2	SHARP_LED1	220
7.46.1.3	SHARP_LED1_DIR	220
7.46.1.4	SHARP_LED2	220
7.46.1.5	SHARP_LED2_DIR	220
7.46.1.6	SHARP_LED3	220
7.46.1.7	SHARP_LED3_DIR	220
7.46.1.8	SHARP_LED4	220
7.46.1.9	SHARP_LED4_DIR	220
7.46.1.10	SHARP_LED5	220
7.46.1.11	SHARP_LED5_DIR	220

---

7.46.1.12 SHARP_VIN . . . . .	220
7.46.1.13 SHARP_VIN_DIR . . . . .	220
7.46.2 Function Documentation . . . . .	220
7.46.2.1 e_get_dist_sharp(void) . . . . .	220
7.46.2.2 e_init_sharp(void) . . . . .	220
7.46.2.3 e_set_sharp_led(unsigned int sharp_led_number, unsigned int value) . . . . .	220
7.46.2.4 e_sharp_led_clear(void) . . . . .	220
7.46.2.5 e_sharp_off(void) . . . . .	220
7.46.2.6 e_sharp_on(void) . . . . .	220
7.47 contrib/SWIS_com_module/ComModule.c File Reference . . . . .	220
7.47.1 Macro Definition Documentation . . . . .	222
7.47.1.1 ADDRHDATA_IN_PACKET_OFFSET . . . . .	222
7.47.1.2 ADDRRLDATA_IN_PACKET_OFFSET . . . . .	222
7.47.1.3 AM_MSGTYPE . . . . .	222
7.47.1.4 AM_MSGTYPE_IN_PACKET_OFFSET . . . . .	222
7.47.1.5 BUFFER_DATA_LENGTH . . . . .	222
7.47.1.6 COM_MODULE_I2C_ADDR . . . . .	222
7.47.1.7 CONFIG_REG_ADDR . . . . .	222
7.47.1.8 FIRSTDATA_IN_PACKET_OFFSET . . . . .	222
7.47.1.9 GROUPDATA_IN_PACKET_OFFSET . . . . .	222
7.47.1.10 HARDWAREATT_SET_FLAG . . . . .	222
7.47.1.11 OWNADDRH_REG_ADDR . . . . .	222
7.47.1.12 OWNADDRL_REG_ADDR . . . . .	222
7.47.1.13 OWNGROUP_REG_ADDR . . . . .	222
7.47.1.14 PACKET_LOST_FLAG . . . . .	222
7.47.1.15 PACKET_READY_FLAG . . . . .	222
7.47.1.16 RADIO_ENABLED_FLAG . . . . .	222
7.47.1.17 REC_BUFFER_END . . . . .	222
7.47.1.18 REC_BUFFER_START . . . . .	222
7.47.1.19 REQUEST_TO_SEND_FLAG . . . . .	222

---

7.47.1.20	SEND_BUFFER_END	222
7.47.1.21	SEND_BUFFER_START	222
7.47.1.22	SEND_REG_ADDR	222
7.47.1.23	SOFTATT_REG_ADDR	223
7.47.1.24	STATUS_REG_ADDR	223
7.47.1.25	TX_IDLE_FLAG	223
7.47.1.26	TX_SEND_ERROR	223
7.47.1.27	TYPEDATA_IN_PACKET_OFFSET	223
7.47.2	Function Documentation	223
7.47.2.1	GetHardwareAttenuator()	223
7.47.2.2	GetOwnAddress()	223
7.47.2.3	GetOwnGroup()	223
7.47.2.4	GetRadioEnabledState()	223
7.47.2.5	GetSoftwareAttenuator()	223
7.47.2.6	GetStatus()	223
7.47.2.7	InitComModule(unsigned char owngroup, unsigned int ownaddress, unsigned char hardwareattenuatormode, unsigned char softwareattenuatorvalue)	223
7.47.2.8	IsModulePlugged()	223
7.47.2.9	IsPacketReady(unsigned char *packet, int *packetsize)	223
7.47.2.10	ReadRegister(unsigned char registeraddr)	223
7.47.2.11	SendPacket(unsigned char destinationgroup, unsigned int destinationaddress, unsigned char *packet, int packetsize)	223
7.47.2.12	SetHardwareAttenuator(unsigned char attenuatormode)	223
7.47.2.13	SetOwnAddress(unsigned int ownaddress)	223
7.47.2.14	SetOwnGroup(unsigned char owngroup)	223
7.47.2.15	SetRadioEnabledState(unsigned char mode)	223
7.47.2.16	SetSoftwareAttenuator(unsigned char attenuatorvalue)	223
7.47.2.17	WriteRegister(unsigned char registeraddr, unsigned char value)	223
7.48	contrib/SWIS_com_module/ComModule.h File Reference	223
7.48.1	Detailed Description	224
7.48.2	Macro Definition Documentation	224

---



7.48.2.1	COM_MODULE_DEFAULT_GROUP	224
7.48.2.2	COM_MODULE_HW_ATTENUATOR_0DB	224
7.48.2.3	COM_MODULE_HW_ATTENUATOR_25DB	224
7.48.2.4	COM_MODULE_MAXSIZE	224
7.48.3	Function Documentation	224
7.48.3.1	GetHardwareAttenuator()	224
7.48.3.2	GetOwnGroup()	225
7.48.3.3	GetRadioEnabledState()	225
7.48.3.4	GetSoftwareAttenuator()	225
7.48.3.5	GetStatus()	225
7.48.3.6	InitComModule(unsigned char owngroup, unsigned int ownaddress, unsigned char hardwareattenuatormode, unsigned char softwareattenuatorvalue)	225
7.48.3.7	IsModulePlugged()	225
7.48.3.8	IsPacketReady(unsigned char *packet, int *packetSize)	225
7.48.3.9	SendPacket(unsigned char destinationgroup, unsigned int destinationaddress, unsigned char *packet, int packetsize)	225
7.48.3.10	SetHardwareAttenuator(unsigned char AttenuatorMode)	225
7.48.3.11	SetOwnAddress(unsigned int ownaddress)	225
7.48.3.12	SetOwnGroup(unsigned char GroupID)	225
7.48.3.13	SetRadioEnabledState(unsigned char mode)	225
7.48.3.14	SetSoftwareAttenuator(unsigned char AttenuatorValue)	225
7.49	fft/e_fft.c File Reference	225
7.49.1	Detailed Description	226
7.49.2	Function Documentation	226
7.49.2.1	__attribute__((space(auto_psv), aligned(FFT_BLOCK_LENGTH *2)))	226
7.49.2.2	e_doFFT_asm(fractcomplex *sigCmpx)	226
7.50	fft/e_fft.h File Reference	226
7.50.1	Detailed Description	226
7.50.2	Macro Definition Documentation	227
7.50.2.1	FFT_BLOCK_LENGTH	227
7.50.2.2	LOG2_BLOCK_LENGTH	227

7.50.3	Function Documentation	227
7.50.3.1	<code>e_doFFT_asm(fractcomplex *sigCmpx)</code>	227
7.51	<code>fft/e_fft_utilities.h</code> File Reference	227
7.51.1	Detailed Description	227
7.51.2	Function Documentation	227
7.51.2.1	<code>e_fast_copy(int *in_array, int *out_array, int size)</code>	227
7.51.2.2	<code>e_subtract_mean(int *array, int size, int log2size)</code>	228
7.52	<code>fft/e_input_signal.c</code> File Reference	228
7.52.1	Detailed Description	228
7.52.2	Function Documentation	229
7.52.2.1	<code>__attribute__((section(".ydata, data, ymemory"), aligned(FFT_BLOCK_LENGTH * 2 * 2)))</code>	229
7.53	<code>fft/e_twiddle_factors.c</code> File Reference	229
7.53.1	Detailed Description	229
7.53.2	Function Documentation	229
7.53.2.1	<code>__attribute__((space(auto_psv), aligned(FFT_BLOCK_LENGTH * 2)))</code>	229
7.54	<code>I2C/e_I2C_master_module.c</code> File Reference	229
7.54.1	Detailed Description	230
7.54.2	Function Documentation	230
7.54.2.1	<code>__attribute__((__interrupt__, auto_psv))</code>	230
7.54.2.2	<code>e_i2c_ack(void)</code>	230
7.54.2.3	<code>e_i2c_deinit(void)</code>	231
7.54.2.4	<code>e_i2c_disable(void)</code>	231
7.54.2.5	<code>e_i2c_enable(void)</code>	231
7.54.2.6	<code>e_i2c_init(void)</code>	231
7.54.2.7	<code>e_i2c_nack(void)</code>	231
7.54.2.8	<code>e_i2c_read(char *buf)</code>	231
7.54.2.9	<code>e_i2c_reset(void)</code>	232
7.54.2.10	<code>e_i2c_restart(void)</code>	232
7.54.2.11	<code>e_i2c_start(void)</code>	232
7.54.2.12	<code>e_i2c_stop(void)</code>	232

---

7.54.2.13	<a href="#">e_i2c_write(char byte)</a>	232
7.54.2.14	<a href="#">idle_i2c(void)</a>	233
7.54.3	<a href="#">Variable Documentation</a>	233
7.54.3.1	<a href="#">e_i2c_mode</a>	233
7.54.3.2	<a href="#">e_interrupts</a>	233
7.55	<a href="#">I2C/e_I2C_master_module.h File Reference</a>	233
7.55.1	<a href="#">Detailed Description</a>	234
7.55.2	<a href="#">Macro Definition Documentation</a>	234
7.55.2.1	<a href="#">ACKNOWLEDGE</a>	234
7.55.2.2	<a href="#">ERROR</a>	234
7.55.2.3	<a href="#">OPERATION_OK</a>	234
7.55.2.4	<a href="#">READ</a>	234
7.55.2.5	<a href="#">RESTART</a>	234
7.55.2.6	<a href="#">START</a>	234
7.55.2.7	<a href="#">STOP</a>	234
7.55.2.8	<a href="#">WRITE</a>	234
7.55.3	<a href="#">Function Documentation</a>	234
7.55.3.1	<a href="#">e_i2c_ack(void)</a>	234
7.55.3.2	<a href="#">e_i2c_deinit(void)</a>	235
7.55.3.3	<a href="#">e_i2c_disable(void)</a>	235
7.55.3.4	<a href="#">e_i2c_enable(void)</a>	235
7.55.3.5	<a href="#">e_i2c_init(void)</a>	235
7.55.3.6	<a href="#">e_i2c_nack(void)</a>	235
7.55.3.7	<a href="#">e_i2c_read(char *buf)</a>	235
7.55.3.8	<a href="#">e_i2c_reset(void)</a>	236
7.55.3.9	<a href="#">e_i2c_restart(void)</a>	236
7.55.3.10	<a href="#">e_i2c_start(void)</a>	236
7.55.3.11	<a href="#">e_i2c_stop(void)</a>	236
7.55.3.12	<a href="#">e_i2c_write(char byte)</a>	236
7.56	<a href="#">I2C/e_I2C_protocol.c File Reference</a>	237

---

7.56.1	Detailed Description	237
7.56.2	Function Documentation	237
7.56.2.1	e_i2cp_deinit(void)	237
7.56.2.2	e_i2cp_disable(void)	237
7.56.2.3	e_i2cp_enable(void)	238
7.56.2.4	e_i2cp_init(void)	238
7.56.2.5	e_i2cp_read(char device_add, char reg)	238
7.56.2.6	e_i2cp_write(char device_add, char reg, char value)	238
7.57	I2C/e_I2C_protocol.h File Reference	239
7.57.1	Detailed Description	239
7.57.2	Function Documentation	239
7.57.2.1	e_i2cp_deinit(void)	239
7.57.2.2	e_i2cp_disable(void)	239
7.57.2.3	e_i2cp_enable(void)	240
7.57.2.4	e_i2cp_init(void)	240
7.57.2.5	e_i2cp_read(char device_add, char reg)	240
7.57.2.6	e_i2cp_read_string(char device_add, unsigned char read_buffer[], char start_↔ address, char string_length)	240
7.57.2.7	e_i2cp_write(char device_add, char reg, char value)	240
7.57.2.8	e_i2cp_write_string(char device_add, unsigned char write_buffer[], char start_↔ address, char string_length)	241
7.58	matlab/matlab files/CloseEpuck.m File Reference	241
7.58.1	Function Documentation	241
7.58.1.1	EpuckPort()	241
7.58.1.2	fclose(EpuckPort)	241
7.58.2	Variable Documentation	241
7.58.2.1	EpuckPort	241
7.59	matlab/matlab files/EpuckFlush.m File Reference	241
7.59.1	Function Documentation	242
7.59.1.1	EpuckPort()	242
7.59.1.2	flushinput(EpuckPort)	242

---

7.59.1.3	flushoutput(EpuckPort) . . . . .	242
7.59.2	Variable Documentation . . . . .	242
7.59.2.1	flush . . . . .	242
7.59.2.2	function . . . . .	242
7.60	matlab/matlab files/EpuckGetData.m File Reference . . . . .	242
7.60.1	Function Documentation . . . . .	243
7.60.1.1	elseif(c== 'C') i . . . . .	243
7.60.1.2	elseif(receivedFormat== 'C') size . . . . .	243
7.60.1.3	if(c~= 'E') continue . . . . .	243
7.60.1.4	if(c== 'l') i . . . . .	243
7.60.1.5	if(receivedFormat== 'l') size . . . . .	243
7.60.1.6	while(i~=5) c . . . . .	243
7.60.2	Variable Documentation . . . . .	243
7.60.2.1	continue . . . . .	243
7.60.2.2	data . . . . .	243
7.60.2.3	function . . . . .	243
7.60.2.4	i . . . . .	243
7.60.2.5	receivedFormat . . . . .	243
7.60.2.6	return . . . . .	243
7.61	matlab/matlab files/EpuckSendData.m File Reference . . . . .	243
7.61.1	Function Documentation . . . . .	244
7.61.1.1	fwrite(EpuckPort, size,'uint16') . . . . .	244
7.61.1.2	fwrite(EpuckPort, data,'int8') . . . . .	244
7.61.1.3	fwrite(EpuckPort, 2 *size,'uint16') . . . . .	244
7.61.1.4	fwrite(EpuckPort, data,'int16') . . . . .	244
7.61.1.5	if(strcmp(dataType,'char')) int8(data) . . . . .	244
7.61.1.6	int16(data) . . . . .	244
7.61.2	Variable Documentation . . . . .	244
7.61.2.1	function . . . . .	244
7.61.2.2	return . . . . .	244

7.61.2.3	size	244
7.62	matlab/matlab files/OpenEpuck.m File Reference	244
7.62.1	Function Documentation	245
7.62.1.1	EpuckPort()	245
7.62.1.2	fopen(EpuckPort)	245
7.62.2	Variable Documentation	245
7.62.2.1	EpuckPort	245
7.62.2.2	Error	245
7.62.2.3	port	245
7.62.2.4	return	245
7.63	matlab/matlab.c File Reference	245
7.63.1	Detailed Description	245
7.63.2	Function Documentation	245
7.63.2.1	e_receive_char_from_matlab(char *data, int array_size)	245
7.63.2.2	e_receive_int_from_matlab(int *data, int array_size)	246
7.63.2.3	e_send_char_to_matlab(char *data, int array_size)	246
7.63.2.4	e_send_int_to_matlab(int *data, int array_size)	246
7.64	matlab/matlab.h File Reference	246
7.64.1	Detailed Description	247
7.64.2	Function Documentation	247
7.64.2.1	e_receive_char_from_matlab(char *data, int array_size)	247
7.64.2.2	e_receive_int_from_matlab(int *data, int array_size)	247
7.64.2.3	e_send_char_to_matlab(char *data, int array_size)	248
7.64.2.4	e_send_int_to_matlab(int *data, int array_size)	248
7.65	motor_led/advance_one_timer/e_agenda.c File Reference	248
7.65.1	Detailed Description	249
7.65.2	Macro Definition Documentation	249
7.65.2.1	EXIT_OK	249
7.65.3	Function Documentation	249
7.65.3.1	__attribute__((interrupt, auto_psv))	249

7.65.3.2	<a href="#">e_activate_agenda(void(*func)(void), int cycle)</a>	249
7.65.3.3	<a href="#">e_destroy_agenda(void(*func)(void))</a>	250
7.65.3.4	<a href="#">e_end_agendas_processing(void)</a>	250
7.65.3.5	<a href="#">e_pause_agenda(void(*func)(void))</a>	250
7.65.3.6	<a href="#">e_reset_agenda(void(*func)(void))</a>	251
7.65.3.7	<a href="#">e_restart_agenda(void(*func)(void))</a>	251
7.65.3.8	<a href="#">e_set_agenda_cycle(void(*func)(void), int cycle)</a>	251
7.65.3.9	<a href="#">e_start_agendas_processing(void)</a>	252
7.65.4	<a href="#">Variable Documentation</a>	252
7.65.4.1	<a href="#">agenda_list</a>	252
7.66	<a href="#">motor_led/advance_one_timer/e_agenda.h File Reference</a>	252
7.66.1	<a href="#">Detailed Description</a>	253
7.66.2	<a href="#">Macro Definition Documentation</a>	253
7.66.2.1	<a href="#">AG_ALREADY_CREATED</a>	253
7.66.2.2	<a href="#">AG_NOT_FOUND</a>	253
7.66.3	<a href="#">Typedef Documentation</a>	253
7.66.3.1	<a href="#">Agenda</a>	253
7.66.4	<a href="#">Function Documentation</a>	253
7.66.4.1	<a href="#">e_activate_agenda(void(*func)(void), int cycle)</a>	253
7.66.4.2	<a href="#">e_destroy_agenda(void(*func)(void))</a>	254
7.66.4.3	<a href="#">e_end_agendas_processing(void)</a>	254
7.66.4.4	<a href="#">e_pause_agenda(void(*func)(void))</a>	255
7.66.4.5	<a href="#">e_reset_agenda(void(*func)(void))</a>	256
7.66.4.6	<a href="#">e_restart_agenda(void(*func)(void))</a>	257
7.66.4.7	<a href="#">e_set_agenda_cycle(void(*func)(void), int cycle)</a>	257
7.66.4.8	<a href="#">e_start_agendas_processing(void)</a>	257
7.67	<a href="#">motor_led/advance_one_timer/e_led.c File Reference</a>	258
7.67.1	<a href="#">Detailed Description</a>	259
7.67.2	<a href="#">Macro Definition Documentation</a>	259
7.67.2.1	<a href="#">LED_EFFECTS</a>	259

7.67.3	Function Documentation	260
7.67.3.1	e_blink_led(void)	260
7.67.3.2	e_blink_led0(void)	260
7.67.3.3	e_blink_led1(void)	260
7.67.3.4	e_blink_led2(void)	260
7.67.3.5	e_blink_led3(void)	261
7.67.3.6	e_blink_led4(void)	261
7.67.3.7	e_blink_led5(void)	261
7.67.3.8	e_blink_led6(void)	261
7.67.3.9	e_blink_led7(void)	261
7.67.3.10	e_led_clear(void)	262
7.67.3.11	e_set_blinking_cycle(int cycle)	262
7.67.3.12	e_set_body_led(unsigned int value)	262
7.67.3.13	e_set_front_led(unsigned int value)	262
7.67.3.14	e_set_led(unsigned int led_number, unsigned int value)	263
7.67.3.15	e_start_led_blinking(int cycle)	263
7.67.3.16	e_stop_led_blinking(void)	263
7.67.3.17	flow_led(void)	263
7.67.3.18	k2000_led(void)	263
7.67.3.19	left_led(void)	264
7.67.3.20	right_led(void)	264
7.67.3.21	snake_led(void)	264
7.68	motor_led/advance_one_timer/fast_agenda/e_led.c File Reference	264
7.68.1	Detailed Description	265
7.68.2	Function Documentation	265
7.68.2.1	e_blink_led(void)	265
7.68.2.2	e_blink_led0(void)	265
7.68.2.3	e_blink_led1(void)	265
7.68.2.4	e_blink_led2(void)	266
7.68.2.5	e_blink_led3(void)	266



---

7.68.2.6	<a href="#">e_blink_led4(void)</a>	266
7.68.2.7	<a href="#">e_blink_led5(void)</a>	266
7.68.2.8	<a href="#">e_blink_led6(void)</a>	266
7.68.2.9	<a href="#">e_blink_led7(void)</a>	267
7.68.2.10	<a href="#">e_led_clear(void)</a>	267
7.68.2.11	<a href="#">e_set_blinking_cycle(int cycle)</a>	267
7.68.2.12	<a href="#">e_set_body_led(unsigned int value)</a>	267
7.68.2.13	<a href="#">e_set_front_led(unsigned int value)</a>	268
7.68.2.14	<a href="#">e_set_led(unsigned int led_number, unsigned int value)</a>	268
7.68.2.15	<a href="#">e_start_led_blinking(int cycle)</a>	268
7.68.2.16	<a href="#">e_stop_led_blinking(void)</a>	268
7.69	<a href="#">motor_led/e_led.c File Reference</a>	269
7.69.1	<a href="#">Detailed Description</a>	269
7.69.2	<a href="#">Function Documentation</a>	269
7.69.2.1	<a href="#">e_led_clear(void)</a>	269
7.69.2.2	<a href="#">e_set_body_led(unsigned int value)</a>	270
7.69.2.3	<a href="#">e_set_front_led(unsigned int value)</a>	271
7.69.2.4	<a href="#">e_set_led(unsigned int led_number, unsigned int value)</a>	271
7.70	<a href="#">motor_led/advance_one_timer/e_led.h File Reference</a>	271
7.70.1	<a href="#">Detailed Description</a>	272
7.70.2	<a href="#">Function Documentation</a>	273
7.70.2.1	<a href="#">e_blink_led(void)</a>	273
7.70.2.2	<a href="#">e_blink_led0(void)</a>	273
7.70.2.3	<a href="#">e_blink_led1(void)</a>	273
7.70.2.4	<a href="#">e_blink_led2(void)</a>	273
7.70.2.5	<a href="#">e_blink_led3(void)</a>	274
7.70.2.6	<a href="#">e_blink_led4(void)</a>	274
7.70.2.7	<a href="#">e_blink_led5(void)</a>	274
7.70.2.8	<a href="#">e_blink_led6(void)</a>	274
7.70.2.9	<a href="#">e_blink_led7(void)</a>	274

---

7.70.2.10	<a href="#">e_led_clear(void)</a>	275
7.70.2.11	<a href="#">e_set_body_led(unsigned int value)</a>	275
7.70.2.12	<a href="#">e_set_front_led(unsigned int value)</a>	275
7.70.2.13	<a href="#">e_set_led(unsigned int led_number, unsigned int value)</a>	275
7.70.2.14	<a href="#">e_start_led_blinking(int cycle)</a>	277
7.70.2.15	<a href="#">e_stop_led_blinking(void)</a>	277
7.70.2.16	<a href="#">flow_led(void)</a>	278
7.70.2.17	<a href="#">k2000_led(void)</a>	278
7.70.2.18	<a href="#">left_led(void)</a>	278
7.70.2.19	<a href="#">right_led(void)</a>	278
7.70.2.20	<a href="#">snake_led(void)</a>	278
7.71	<a href="#">motor_led/advance_one_timer/fast_agenda/e_led.h File Reference</a>	278
7.71.1	<a href="#">Detailed Description</a>	279
7.71.2	<a href="#">Function Documentation</a>	279
7.71.2.1	<a href="#">e_blink_led(void)</a>	279
7.71.2.2	<a href="#">e_blink_led0(void)</a>	279
7.71.2.3	<a href="#">e_blink_led1(void)</a>	280
7.71.2.4	<a href="#">e_blink_led2(void)</a>	280
7.71.2.5	<a href="#">e_blink_led3(void)</a>	280
7.71.2.6	<a href="#">e_blink_led4(void)</a>	280
7.71.2.7	<a href="#">e_blink_led5(void)</a>	280
7.71.2.8	<a href="#">e_blink_led6(void)</a>	281
7.71.2.9	<a href="#">e_blink_led7(void)</a>	281
7.71.2.10	<a href="#">e_led_clear(void)</a>	281
7.71.2.11	<a href="#">e_set_body_led(unsigned int value)</a>	281
7.71.2.12	<a href="#">e_set_front_led(unsigned int value)</a>	282
7.71.2.13	<a href="#">e_set_led(unsigned int led_number, unsigned int value)</a>	282
7.71.2.14	<a href="#">e_start_led_blinking(int cycle)</a>	282
7.71.2.15	<a href="#">e_stop_led_blinking(void)</a>	283
7.72	<a href="#">motor_led/e_led.h File Reference</a>	283

---

---

7.72.1	Detailed Description	283
7.72.2	Function Documentation	284
7.72.2.1	e_led_clear(void)	284
7.72.2.2	e_set_body_led(unsigned int value)	284
7.72.2.3	e_set_front_led(unsigned int value)	284
7.72.2.4	e_set_led(unsigned int led_number, unsigned int value)	285
7.73	motor_led/advance_one_timer/e_motors.c File Reference	285
7.73.1	Detailed Description	286
7.73.2	Macro Definition Documentation	286
7.73.2.1	MAXV	286
7.73.2.2	POWERSAVE	286
7.73.2.3	TRESHV	286
7.73.3	Function Documentation	286
7.73.3.1	e_get_steps_left()	286
7.73.3.2	e_get_steps_right()	287
7.73.3.3	e_init_motors(void)	287
7.73.3.4	e_set_speed(int linear_speed, int angular_speed)	287
7.73.3.5	e_set_speed_left(int motor_speed)	287
7.73.3.6	e_set_speed_right(int motor_speed)	288
7.73.3.7	e_set_steps_left(int set_steps)	288
7.73.3.8	e_set_steps_right(int set_steps)	288
7.73.3.9	run_left_motor(void)	288
7.73.3.10	run_right_motor(void)	288
7.73.4	Variable Documentation	289
7.73.4.1	left_motor_phase	289
7.73.4.2	left_speed	289
7.73.4.3	nbr_steps_left	289
7.73.4.4	nbr_steps_right	289
7.73.4.5	right_motor_phase	289
7.73.4.6	right_speed	289

7.74 motor_led/advance_one_timer/fast_agenda/e_motors.c File Reference . . . . .	289
7.74.1 Detailed Description . . . . .	290
7.74.2 Macro Definition Documentation . . . . .	290
7.74.2.1 MAXV . . . . .	290
7.74.2.2 POWERSAVE . . . . .	290
7.74.2.3 TRESHV . . . . .	290
7.74.3 Function Documentation . . . . .	290
7.74.3.1 e_get_steps_left() . . . . .	290
7.74.3.2 e_get_steps_right() . . . . .	290
7.74.3.3 e_init_motors(void) . . . . .	291
7.74.3.4 e_set_speed(int linear_speed, int angular_speed) . . . . .	291
7.74.3.5 e_set_speed_left(int motor_speed) . . . . .	291
7.74.3.6 e_set_speed_right(int motor_speed) . . . . .	291
7.74.3.7 e_set_steps_left(int set_steps) . . . . .	292
7.74.3.8 e_set_steps_right(int set_steps) . . . . .	292
7.74.3.9 run_left_motor(void) . . . . .	292
7.74.3.10 run_right_motor(void) . . . . .	292
7.74.4 Variable Documentation . . . . .	292
7.74.4.1 left_motor_phase . . . . .	292
7.74.4.2 left_speed . . . . .	292
7.74.4.3 nbr_steps_left . . . . .	292
7.74.4.4 nbr_steps_right . . . . .	292
7.74.4.5 right_motor_phase . . . . .	293
7.74.4.6 right_speed . . . . .	293
7.75 motor_led/e_motors.c File Reference . . . . .	293
7.75.1 Detailed Description . . . . .	294
7.75.2 Function Documentation . . . . .	294
7.75.2.1 __attribute__((interrupt, auto_psv, shadow)) . . . . .	294
7.75.2.2 e_get_steps_left(void) . . . . .	294
7.75.2.3 e_get_steps_right(void) . . . . .	294

---

7.75.2.4	<a href="#">e_init_motors(void)</a>	295
7.75.2.5	<a href="#">e_set_speed_left(int motor_speed)</a>	295
7.75.2.6	<a href="#">e_set_speed_right(int motor_speed)</a>	295
7.75.2.7	<a href="#">e_set_steps_left(int set_steps)</a>	295
7.75.2.8	<a href="#">e_set_steps_right(int set_steps)</a>	296
7.75.3	<a href="#">Variable Documentation</a>	296
7.75.3.1	<a href="#">left_speed</a>	296
7.75.3.2	<a href="#">nbr_pas_left</a>	296
7.75.3.3	<a href="#">nbr_pas_right</a>	296
7.75.3.4	<a href="#">right_speed</a>	296
7.76	<a href="#">motor_led/advance_one_timer/e_motors.h File Reference</a>	296
7.76.1	<a href="#">Detailed Description</a>	297
7.76.2	<a href="#">Function Documentation</a>	297
7.76.2.1	<a href="#">e_get_steps_left()</a>	297
7.76.2.2	<a href="#">e_get_steps_right()</a>	297
7.76.2.3	<a href="#">e_init_motors(void)</a>	298
7.76.2.4	<a href="#">e_set_speed(int linear_speed, int angular_speed)</a>	298
7.76.2.5	<a href="#">e_set_speed_left(int motor_speed)</a>	298
7.76.2.6	<a href="#">e_set_speed_right(int motor_speed)</a>	299
7.76.2.7	<a href="#">e_set_steps_left(int steps_left)</a>	300
7.76.2.8	<a href="#">e_set_steps_right(int steps_right)</a>	301
7.77	<a href="#">motor_led/advance_one_timer/fast_agenda/e_motors.h File Reference</a>	301
7.77.1	<a href="#">Detailed Description</a>	301
7.77.2	<a href="#">Function Documentation</a>	302
7.77.2.1	<a href="#">e_get_steps_left()</a>	302
7.77.2.2	<a href="#">e_get_steps_right()</a>	302
7.77.2.3	<a href="#">e_init_motors(void)</a>	302
7.77.2.4	<a href="#">e_set_speed(int linear_speed, int angular_speed)</a>	302
7.77.2.5	<a href="#">e_set_speed_left(int motor_speed)</a>	303
7.77.2.6	<a href="#">e_set_speed_right(int motor_speed)</a>	304

---

7.77.2.7	<code>e_set_steps_left(int steps_left)</code>	305
7.77.2.8	<code>e_set_steps_right(int steps_right)</code>	305
7.78	<code>motor_led/e_motors.h</code> File Reference	305
7.78.1	Detailed Description	306
7.78.2	Function Documentation	306
7.78.2.1	<code>e_get_steps_left(void)</code>	306
7.78.2.2	<code>e_get_steps_right(void)</code>	306
7.78.2.3	<code>e_init_motors(void)</code>	307
7.78.2.4	<code>e_set_speed_left(int motor_speed)</code>	307
7.78.2.5	<code>e_set_speed_right(int motor_speed)</code>	308
7.78.2.6	<code>e_set_steps_left(int set_steps)</code>	309
7.78.2.7	<code>e_set_steps_right(int set_steps)</code>	309
7.79	<code>motor_led/advance_one_timer/e_remote_control.c</code> File Reference	310
7.79.1	Detailed Description	310
7.79.2	Function Documentation	311
7.79.2.1	<code>__attribute__((__interrupt__, auto_psv))</code>	311
7.79.2.2	<code>e_get_address(void)</code>	311
7.79.2.3	<code>e_get_check(void)</code>	311
7.79.2.4	<code>e_get_data(void)</code>	311
7.79.2.5	<code>e_init_remote_control(void)</code>	311
7.79.2.6	<code>e_read_remote_control(void)</code>	311
7.79.3	Variable Documentation	312
7.79.3.1	<code>address</code>	312
7.79.3.2	<code>address_temp</code>	312
7.79.3.3	<code>check</code>	312
7.79.3.4	<code>check_temp</code>	312
7.79.3.5	<code>data</code>	312
7.79.3.6	<code>data_temp</code>	312
7.80	<code>motor_led/advance_one_timer/e_remote_control.h</code> File Reference	312
7.80.1	Detailed Description	313

---

---

7.80.2	Macro Definition Documentation	314
7.80.2.1	BOTTOMI	314
7.80.2.2	BOTTOMR	314
7.80.2.3	CHAN_DOWN	314
7.80.2.4	CHAN_UP	314
7.80.2.5	I_II	314
7.80.2.6	MUTE	314
7.80.2.7	OUT_AUX_1	314
7.80.2.8	STANDBY	314
7.80.2.9	VOL_DOWN	314
7.80.2.10	VOL_UP	314
7.80.3	Function Documentation	314
7.80.3.1	e_get_address(void)	314
7.80.3.2	e_get_check(void)	314
7.80.3.3	e_get_data(void)	314
7.80.3.4	e_init_remote_control(void)	315
7.80.3.5	e_read_remote_control(void)	315
7.81	motor_led/advance_one_timer/fast_agenda/e_agenda_fast.c File Reference	315
7.81.1	Detailed Description	316
7.81.2	Macro Definition Documentation	316
7.81.2.1	EXIT_OK	316
7.81.3	Function Documentation	316
7.81.3.1	__attribute__((__interrupt__, auto_psv))	316
7.81.3.2	assign_agenda(Agenda *agenda)	316
7.81.3.3	compute_gcd(unsigned u, unsigned v)	316
7.81.3.4	e_activate_agenda(void(*func)(void), unsigned cycle)	316
7.81.3.5	e_activate_motors(void(*func1)(void), void(*func2)(void))	317
7.81.3.6	e_configure_timer(int timer)	317
7.81.3.7	e_destroy_agenda(void(*func)(void))	317
7.81.3.8	e_end_agendas_processing(int timer)	317

7.81.3.9	<code>e_pause_agenda(void(*func)(void))</code>	318
7.81.3.10	<code>e_reset_agenda(void(*func)(void))</code>	318
7.81.3.11	<code>e_restart_agenda(void(*func)(void))</code>	318
7.81.3.12	<code>e_set_agenda_cycle(void(*func)(void), unsigned cycle)</code>	318
7.81.3.13	<code>e_set_motor_speed(void(*func)(void), unsigned cycle)</code>	318
7.81.3.14	<code>e_set_timer_speed(int timer, unsigned speed)</code>	318
7.81.3.15	<code>e_start_agendas_processing(void)</code>	318
7.81.3.16	<code>e_start_timer_processing(int timer)</code>	319
7.81.3.17	<code>find_function(void(*func)(void))</code>	319
7.81.3.18	<code>migrate(int timer)</code>	319
7.81.3.19	<code>my_ceil(float a)</code>	319
7.81.3.20	<code>recompute_speeds()</code>	319
7.81.3.21	<code>search_best_fit(unsigned cycle)</code>	319
7.81.4	Variable Documentation	319
7.81.4.1	<code>agenda_list</code>	319
7.82	<code>motor_led/advance_one_timer/fast_agenda/e_agenda_fast.h</code> File Reference	319
7.82.1	Detailed Description	320
7.82.2	Macro Definition Documentation	321
7.82.2.1	<code>AG_ALREADY_CREATED</code>	321
7.82.2.2	<code>AG_NOT_FOUND</code>	321
7.82.3	Typedef Documentation	321
7.82.3.1	<code>Agenda</code>	321
7.82.4	Function Documentation	321
7.82.4.1	<code>e_activate_agenda(void(*func)(void), unsigned cycle)</code>	321
7.82.4.2	<code>e_activate_motors(void(*func1)(void), void(*func2)(void))</code>	321
7.82.4.3	<code>e_configure_timer(int timer)</code>	321
7.82.4.4	<code>e_destroy_agenda(void(*func)(void))</code>	321
7.82.4.5	<code>e_end_agendas_processing(int timer)</code>	322
7.82.4.6	<code>e_pause_agenda(void(*func)(void))</code>	322
7.82.4.7	<code>e_reset_agenda(void(*func)(void))</code>	323



---

7.82.4.8	<a href="#">e_restart_agenda(void(*func)(void))</a>	323
7.82.4.9	<a href="#">e_set_agenda_cycle(void(*func)(void), unsigned cycle)</a>	324
7.82.4.10	<a href="#">e_set_motor_speed(void(*func)(void), unsigned cycle)</a>	324
7.82.4.11	<a href="#">e_start_agendas_processing(void)</a>	324
7.82.4.12	<a href="#">e_start_timer_processing(int timer)</a>	324
7.83	<a href="#">motor_led/e_epuck_ports.h File Reference</a>	324
7.83.1	<a href="#">Detailed Description</a>	327
7.83.2	<a href="#">Macro Definition Documentation</a>	327
7.83.2.1	<a href="#">ACCX</a>	327
7.83.2.2	<a href="#">ACCY</a>	327
7.83.2.3	<a href="#">ACCZ</a>	327
7.83.2.4	<a href="#">AUDIO_ON</a>	327
7.83.2.5	<a href="#">AUDIO_ON_DIR</a>	327
7.83.2.6	<a href="#">BATT_LOW</a>	327
7.83.2.7	<a href="#">BATT_LOW_DIR</a>	327
7.83.2.8	<a href="#">BODY_LED</a>	327
7.83.2.9	<a href="#">BODY_LED_DIR</a>	327
7.83.2.10	<a href="#">CAM_DATA</a>	327
7.83.2.11	<a href="#">CAM_HREF</a>	327
7.83.2.12	<a href="#">CAM_HREF_DIR</a>	328
7.83.2.13	<a href="#">CAM_PCLK</a>	328
7.83.2.14	<a href="#">CAM_PCLK_DIR</a>	328
7.83.2.15	<a href="#">CAM_PWDN</a>	328
7.83.2.16	<a href="#">CAM_PWDN_DIR</a>	328
7.83.2.17	<a href="#">CAM_RESET</a>	328
7.83.2.18	<a href="#">CAM_RESET_DIR</a>	328
7.83.2.19	<a href="#">CAM_VSYNC</a>	328
7.83.2.20	<a href="#">CAM_VSYNC_DIR</a>	328
7.83.2.21	<a href="#">CAM_y0</a>	328
7.83.2.22	<a href="#">CAM_y0_DIR</a>	328

---

7.83.2.23 CAM_y1 . . . . .	328
7.83.2.24 CAM_y1_DIR . . . . .	328
7.83.2.25 CAM_y2 . . . . .	328
7.83.2.26 CAM_y2_DIR . . . . .	328
7.83.2.27 CAM_y3 . . . . .	328
7.83.2.28 CAM_y3_DIR . . . . .	328
7.83.2.29 CAM_y4 . . . . .	328
7.83.2.30 CAM_y4_DIR . . . . .	328
7.83.2.31 CAM_y5 . . . . .	328
7.83.2.32 CAM_y5_DIR . . . . .	328
7.83.2.33 CAM_y6 . . . . .	328
7.83.2.34 CAM_y6_DIR . . . . .	328
7.83.2.35 CAM_y7 . . . . .	329
7.83.2.36 CAM_y7_DIR . . . . .	329
7.83.2.37 CLRWDT . . . . .	329
7.83.2.38 FALSE . . . . .	329
7.83.2.39 FCY . . . . .	329
7.83.2.40 FOSC . . . . .	329
7.83.2.41 FRONT_LED . . . . .	329
7.83.2.42 FRONT_LED_DIR . . . . .	329
7.83.2.43 IDLE . . . . .	329
7.83.2.44 INPUT_PIN . . . . .	329
7.83.2.45 INTERRUPT_DELAY . . . . .	329
7.83.2.46 INTERRUPT_OFF . . . . .	329
7.83.2.47 INTERRUPT_ON . . . . .	329
7.83.2.48 IR0 . . . . .	329
7.83.2.49 IR1 . . . . .	329
7.83.2.50 IR2 . . . . .	329
7.83.2.51 IR3 . . . . .	329
7.83.2.52 IR4 . . . . .	329

---

---

7.83.2.53 IR5 . . . . .	329
7.83.2.54 IR6 . . . . .	329
7.83.2.55 IR7 . . . . .	329
7.83.2.56 LED0 . . . . .	329
7.83.2.57 LED0_DIR . . . . .	329
7.83.2.58 LED1 . . . . .	330
7.83.2.59 LED1_DIR . . . . .	330
7.83.2.60 LED2 . . . . .	330
7.83.2.61 LED2_DIR . . . . .	330
7.83.2.62 LED3 . . . . .	330
7.83.2.63 LED3_DIR . . . . .	330
7.83.2.64 LED4 . . . . .	330
7.83.2.65 LED4_DIR . . . . .	330
7.83.2.66 LED5 . . . . .	330
7.83.2.67 LED5_DIR . . . . .	330
7.83.2.68 LED6 . . . . .	330
7.83.2.69 LED6_DIR . . . . .	330
7.83.2.70 LED7 . . . . .	330
7.83.2.71 LED7_DIR . . . . .	330
7.83.2.72 MIC1 . . . . .	330
7.83.2.73 MIC2 . . . . .	330
7.83.2.74 MIC3 . . . . .	330
7.83.2.75 MICROSEC . . . . .	330
7.83.2.76 MILLISEC . . . . .	330
7.83.2.77 MOTOR1_PHA . . . . .	330
7.83.2.78 MOTOR1_PHA_DIR . . . . .	330
7.83.2.79 MOTOR1_PHB . . . . .	330
7.83.2.80 MOTOR1_PHB_DIR . . . . .	330
7.83.2.81 MOTOR1_PHC . . . . .	331
7.83.2.82 MOTOR1_PHC_DIR . . . . .	331

---

7.83.2.83 MOTOR1_PHD . . . . .	331
7.83.2.84 MOTOR1_PHD_DIR . . . . .	331
7.83.2.85 MOTOR2_PHA . . . . .	331
7.83.2.86 MOTOR2_PHA_DIR . . . . .	331
7.83.2.87 MOTOR2_PHB . . . . .	331
7.83.2.88 MOTOR2_PHB_DIR . . . . .	331
7.83.2.89 MOTOR2_PHC . . . . .	331
7.83.2.90 MOTOR2_PHC_DIR . . . . .	331
7.83.2.91 MOTOR2_PHD . . . . .	331
7.83.2.92 MOTOR2_PHD_DIR . . . . .	331
7.83.2.93 NANOSEC . . . . .	331
7.83.2.94 NOP . . . . .	331
7.83.2.95 OUTPUT_PIN . . . . .	331
7.83.2.96 PLL . . . . .	331
7.83.2.97 PULSE_IR0 . . . . .	331
7.83.2.98 PULSE_IR0_DIR . . . . .	331
7.83.2.99 PULSE_IR1 . . . . .	331
7.83.2.100PULSE_IR1_DIR . . . . .	331
7.83.2.101PULSE_IR2 . . . . .	331
7.83.2.102PULSE_IR2_DIR . . . . .	331
7.83.2.103PULSE_IR3 . . . . .	331
7.83.2.104PULSE_IR3_DIR . . . . .	332
7.83.2.105REMOTE . . . . .	332
7.83.2.106REMOTE_DIR . . . . .	332
7.83.2.107RESET . . . . .	332
7.83.2.108SELECTOR0 . . . . .	332
7.83.2.109SELECTOR0_DIR . . . . .	332
7.83.2.110SELECTOR1 . . . . .	332
7.83.2.111SELECTOR1_DIR . . . . .	332
7.83.2.112SELECTOR2 . . . . .	332

---

---

7.83.2.113	SELECTOR2_DIR . . . . .	332
7.83.2.114	SELECTOR3 . . . . .	332
7.83.2.115	SELECTOR3_DIR . . . . .	332
7.83.2.116	SIO_C . . . . .	332
7.83.2.117	SIO_C_DIR . . . . .	332
7.83.2.118	SIO_D . . . . .	332
7.83.2.119	SIO_D_DIR . . . . .	332
7.83.2.120	SLEEP . . . . .	332
7.83.2.121	STOP_TMR1 . . . . .	332
7.83.2.122	STOP_TMR2 . . . . .	332
7.83.2.123	STOP_TMR3 . . . . .	332
7.83.2.124	STOP_TMR4 . . . . .	332
7.83.2.125	STOP_TMR5 . . . . .	332
7.83.2.126	TCY_PIC . . . . .	332
7.83.2.127	TRUE . . . . .	332
7.84	motor_led/e_init_port.c File Reference . . . . .	332
7.84.1	Detailed Description . . . . .	333
7.84.2	Function Documentation . . . . .	333
7.84.2.1	_FBORPOR(PBOR_OFF &MCLR_EN) . . . . .	333
7.84.2.2	_FGS(CODE_PROT_OFF) . . . . .	333
7.84.2.3	_FOSC(CSW_FSCM_OFF &XT_PLL8) . . . . .	333
7.84.2.4	_FWDT(WDT_OFF) . . . . .	333
7.84.2.5	e_init_port(void) . . . . .	333
7.84.2.6	isEpuckVersion1_3(void) . . . . .	334
7.84.2.7	testAccGyroPresence() . . . . .	334
7.84.3	Variable Documentation . . . . .	334
7.84.3.1	isPresentFlag . . . . .	334
7.85	motor_led/e_init_port.h File Reference . . . . .	334
7.85.1	Detailed Description . . . . .	334
7.85.2	Function Documentation . . . . .	334

---

7.85.2.1	<code>e_init_port(void)</code> . . . . .	334
7.85.2.2	<code>isEpuckVersion1_3(void)</code> . . . . .	334
7.86	<code>motor_led/e_motors_timer3.c</code> File Reference . . . . .	334
7.86.1	Detailed Description . . . . .	335
7.86.2	Function Documentation . . . . .	336
7.86.2.1	<code>__attribute__((interrupt, auto_psv, shadow))</code> . . . . .	336
7.86.2.2	<code>e_get_steps_left(void)</code> . . . . .	336
7.86.2.3	<code>e_get_steps_right(void)</code> . . . . .	336
7.86.2.4	<code>e_init_motors(void)</code> . . . . .	336
7.86.2.5	<code>e_set_speed_left(int motor_speed)</code> . . . . .	336
7.86.2.6	<code>e_set_speed_right(int motor_speed)</code> . . . . .	337
7.86.2.7	<code>e_set_steps_left(int set_steps)</code> . . . . .	337
7.86.2.8	<code>e_set_steps_right(int set_steps)</code> . . . . .	337
7.86.3	Variable Documentation . . . . .	337
7.86.3.1	<code>left_speed</code> . . . . .	337
7.86.3.2	<code>motor_counter_left</code> . . . . .	337
7.86.3.3	<code>motor_counter_left_init</code> . . . . .	337
7.86.3.4	<code>motor_counter_right</code> . . . . .	337
7.86.3.5	<code>motor_counter_right_init</code> . . . . .	337
7.86.3.6	<code>nbr_pas_left</code> . . . . .	337
7.86.3.7	<code>nbr_pas_right</code> . . . . .	338
7.86.3.8	<code>right_speed</code> . . . . .	338
7.87	<code>uart/e_epuck_ports.inc</code> File Reference . . . . .	338
7.88	<code>uart/e_uart_char.h</code> File Reference . . . . .	338
7.88.1	Detailed Description . . . . .	339
7.88.2	Macro Definition Documentation . . . . .	339
7.88.2.1	<code>BAUD115200</code> . . . . .	339
7.88.2.2	<code>BAUD230400</code> . . . . .	339
7.88.2.3	<code>BAUD460800</code> . . . . .	339
7.88.2.4	<code>BAUD921600</code> . . . . .	339

7.88.3	Function Documentation	339
7.88.3.1	e_getchar_uart1(char *car)	339
7.88.3.2	e_getchar_uart2(char *car)	340
7.88.3.3	e_init_uart1(void)	340
7.88.3.4	e_init_uart2(int)	340
7.88.3.5	e_ischar_uart1()	340
7.88.3.6	e_ischar_uart2()	340
7.88.3.7	e_send_uart1_char(const char *buff, int length)	340
7.88.3.8	e_send_uart2_char(const char *buff, int length)	341
7.88.3.9	e_uart1_sending(void)	341
7.88.3.10	e_uart2_sending(void)	341
7.88.4	Variable Documentation	341
7.88.4.1	e_uart1_int_clr_addr	341
7.88.4.2	e_uart1_int_clr_mask	341
7.88.4.3	e_uart2_int_clr_addr	341
7.88.4.4	e_uart2_int_clr_mask	341
7.89	utility/utility.c File Reference	341
7.89.1	Macro Definition Documentation	342
7.89.1.1	BATT_VALUES_RANGE	342
7.89.1.2	MAX_BATT_VALUE	342
7.89.1.3	MIN_BATT_VALUE	342
7.89.2	Function Documentation	342
7.89.2.1	getBatteryValuePercentage()	342
7.89.2.2	getBatteryValueRaw()	342
7.89.2.3	getDiffTimeMs(void)	342
7.89.2.4	getDiffTimeMsAndReset(void)	342
7.89.2.5	getselector()	342
7.89.2.6	resetTime(void)	342
7.89.2.7	wait(long num)	342
7.89.3	Variable Documentation	342
7.89.3.1	e_acc_scan	342
7.89.3.2	e_last_acc_scan_id	343
7.89.3.3	tickAdclsr	343
7.90	utility/utility.h File Reference	343
7.90.1	Function Documentation	343
7.90.1.1	getBatteryValuePercentage()	343
7.90.1.2	getBatteryValueRaw()	343
7.90.1.3	getDiffTimeMs(void)	343
7.90.1.4	getDiffTimeMsAndReset(void)	343
7.90.1.5	getselector()	343
7.90.1.6	resetTime(void)	343
7.90.1.7	wait(long num)	343

[Index](#)

345



# Chapter 1

## e-puck standard library documentation

### 1.1 Introduction

This project has been started at the Ecole Polytechnique Federale de Lausanne as collaboration between the Autonomous Systems Lab, the Swarm-Intelligent Systems group and the Laboratory of Intelligent System.

An educational robot: The main goal of this project is to develop a miniature mobile robot for educational purposes at university level. To achieve this goal the robot needs, in our opinion, the following features:

- Good structure. The robot should have a clean mechanical structure, simple to understand. The electronics, processor structure and software have to be a good example of a clean modern system.
- Flexibility. The robot should cover a large spectrum of educational activities and should therefore have a large potential in its sensors, processing power and extensions. Potential educational fields are, for instance, mobile robotics, real-time programming, embedded systems, signal processing, image or sound feature extraction, human-machine interaction or collective systems.
- User friendly. The robot should be small and easy to exploit on a table next to a computer. It should need minimal wiring, battery operation and optimal working comfort.
- Good robustness and simple maintenance. The robot should resist to student use and be simple and cheap to repair.
- Cheap. The robot, for large use, should be cheap (450-550 euros)

### 1.2 Documentation organization

This documentation is divided in five sections (as you can see on the top of the page):

- Main Page: The startup page.
- Modules: An overview of all the modules that compose this library. Here you can see all the files containing by each module and a detailed description of each module. Look at these pages to have a better idea of what each module is doing.
- Data Structures: Here are listed all the C-struct of the library.
- Files: All the library's files listed by alphabetical order.
- Directories: The directories architectures of the library.

### 1.3 External links

- <http://www.e-puck.org/> The official site of the e-puck
- <https://gna.org/projects/e-puck/> The developers area at gna
- <http://lsro.epfl.ch/> The site of the lab where the e-puck has been created
- [http://www.e-puck.org/index.php?option=com\\_content&task=view&id=18&Itemid=45](http://www.e-puck.org/index.php?option=com_content&task=view&id=18&Itemid=45) The license

# Chapter 2

## Module Index

### 2.1 Modules

Here is a list of all modules:

Analogic/Digital conversion (ADC)	11
Bluetooth	13
Camera fast two timers	14
Camera slow three timers	17
Sound	19
LIS sensor turret	21
Radio communication	22
FFT	23
I2C	24
Matlab communication	27
Ports, motors and LEDs	28
UART	31



# Chapter 3

## Data Structure Index

### 3.1 Data Structures

Here are the data structures with brief descriptions:

- [AgendaList](#)
  - Manage the differents agendas lists . . . . . 33
- [AgendaType](#)
  - Srtuct Agenda as chained list . . . . . 34
- [BtDevice](#)
  - General struct for bluetooth device . . . . . 35
- [BtEPuck](#)
  - General struct for other e-puck . . . . . 36
- [TypeAccRaw](#)
  - Struct to store the acceleration raw data in carthesian coord . . . . . 37
- [TypeAccSpheric](#)
  - Struct to store the acceleration vector in spherical coord . . . . . 37



# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

<a href="#">a_d/e_accelerometer.c</a>	Accessing the accelerometer sensor data	75
<a href="#">a_d/e_accelerometer.h</a>	Accessing the accelerometer sensor data	76
<a href="#">a_d/e_ad_conv.c</a>	Module for the Analogic/Digital conversion	52
<a href="#">a_d/e_ad_conv.h</a>	Module for the Analogic/Digital conversion	56
<a href="#">a_d/e_micro.c</a>	Accessing the microphone data	59
<a href="#">a_d/e_micro.h</a>	Accessing the microphone data	62
<a href="#">a_d/e_prox.c</a>	Accessing proximity sensor of e-puck with timer 1	66
<a href="#">a_d/e_prox.h</a>	Accessing proximity sensor of e-puck with timer 1	72
<a href="#">a_d/e_prox_timer2.c</a>	Control proximity sensor of e-puck with timer2	77
<a href="#">a_d/advance_ad_scan/e_acc.c</a>	Accessing the accelerometer data	39
<a href="#">a_d/advance_ad_scan/e_acc.h</a>	Accessing the accelerometer data	44
<a href="#">a_d/advance_ad_scan/e_ad_conv.c</a>	Module for the advance Analogic/Digital conversion	49
<a href="#">a_d/advance_ad_scan/e_ad_conv.h</a>	Module for the advance Analogic/Digital conversion	53
<a href="#">a_d/advance_ad_scan/e_micro.c</a>	Accessing the microphone data	57
<a href="#">a_d/advance_ad_scan/e_micro.h</a>	Accessing the microphone data	60
<a href="#">a_d/advance_ad_scan/e_prox.c</a>	Accessing proximity sensor of e-puck	63
<a href="#">a_d/advance_ad_scan/e_prox.h</a>	Accessing proximity sensor of e-puck	69
<a href="#">acc_gyro/e_lsm330.c</a>	Manage LSM330 (accelerometer + gyro) registers	79

acc_gyro/e_lsm330.h	
LSM330 library header	82
bluetooth/e_bluetooth.c	
Manage Bluetooth	84
bluetooth/e_bluetooth.h	
Manage Bluetooth	91
camera/fast_2_timer/e_calc_po3030k.c	
Calculate the timing for the camera (two timers)	98
camera/fast_2_timer/e_calc_po6030k.c	
Calculate the timing for the camera	100
camera/fast_2_timer/e_calc_po8030d.c	
Calculate the timing for the camera	102
camera/fast_2_timer/e_common.c	103
camera/fast_2_timer/e_po3030k.h	
PO3030k library header (two timers)	105
camera/fast_2_timer/e_po3030k_registers.c	
Manage po3030k registers (two timers)	140
camera/fast_2_timer/e_po6030k.h	
PO6030k library header	157
camera/fast_2_timer/e_po6030k_registers.c	
Manage po6030k registers (two timers)	165
camera/fast_2_timer/e_po8030d.h	
PO8030d library header	170
camera/fast_2_timer/e_po8030d_registers.c	
Manage po8030d registers (two timers)	176
camera/fast_2_timer/e_poxxxx.h	
Camera library header	181
camera/fast_2_timer/e_timers.c	
Manage camera's interrupts (two timers)	185
camera/slow_3_timer/e_calc.c	
Calculate the timing for the camera (three timers)	192
camera/slow_3_timer/e_po3030k.h	
PO3030k library header (three timers)	122
camera/slow_3_timer/e_registers.c	
Manage po3030k registers (three timers)	193
camera/slow_3_timer/e_timers.c	
Manage camera's interrupts (three timers)	188
codec/e_common.inc	211
codec/e_sound.c	
Package to play basics sounds on the e-puck's speaker.	
For more info look at this: <a href="#">Sound</a>	211
codec/e_sound.h	
Package to play basics sounds on the e-puck's speaker	212
contrib/LIS_sensors_turret/e_devantech.c	214
contrib/LIS_sensors_turret/e_devantech.h	
Devantech sensor of e-puck	215
contrib/LIS_sensors_turret/e_sensex.c	216
contrib/LIS_sensors_turret/e_sensex.h	217
contrib/LIS_sensors_turret/e_sharp.c	218
contrib/LIS_sensors_turret/e_sharp.h	219
contrib/SWIS_com_module/ComModule.c	220
contrib/SWIS_com_module/ComModule.h	
Radio communication	223
fft/e_fft.c	
Package to manage the FFT	225
fft/e_fft.h	
Package to manage the FFT	226



<a href="#">fft/e_fft_utilities.h</a>	
Some fft features	227
<a href="#">fft/e_input_signal.c</a>	
Allocate memory and initialize the sigCmpx array	228
<a href="#">fft/e_twiddle_factors.c</a>	
The FFT factor from Microchip	229
<a href="#">I2C/e_I2C_master_module.c</a>	
Manage I2C basics	229
<a href="#">I2C/e_I2C_master_module.h</a>	
Manage I2C basics	233
<a href="#">I2C/e_I2C_protocol.c</a>	
Manage I2C protocole	237
<a href="#">I2C/e_I2C_protocol.h</a>	
Manage I2C protocole	239
<a href="#">matlab/matlab.c</a>	
To communicate with matlab	245
<a href="#">matlab/matlab.h</a>	
To communicate with matlab	246
<a href="#">matlab/matlab files/CloseEpuck.m</a>	241
<a href="#">matlab/matlab files/EpuckFlush.m</a>	241
<a href="#">matlab/matlab files/EpuckGetData.m</a>	242
<a href="#">matlab/matlab files/EpuckSendData.m</a>	243
<a href="#">matlab/matlab files/OpenEpuck.m</a>	244
<a href="#">motor_led/e_epuck_ports.h</a>	
Define all the usefull names corresponding of e-puck's hardware	324
<a href="#">motor_led/e_init_port.c</a>	
Initialize the ports on standard configuration	332
<a href="#">motor_led/e_init_port.h</a>	
Initialize the ports on standard configuration	334
<a href="#">motor_led/e_led.c</a>	
Manage the LEDs.	
A little exemple for the LEDs (all the LEDs are blinking)	269
<a href="#">motor_led/e_led.h</a>	
Manage the LEDs.	
A little exemple for the LEDs (all the LEDs are blinking)	283
<a href="#">motor_led/e_motors.c</a>	
Manage the motors (with timer 4 and 5)	293
<a href="#">motor_led/e_motors.h</a>	
Manage the motors (with timer 4 and 5)	305
<a href="#">motor_led/e_motors_timer3.c</a>	
Initialize the ports on standard configuration	334
<a href="#">motor_led/advance_one_timer/e_agenda.c</a>	
Manage the agendas (timer2)	248
<a href="#">motor_led/advance_one_timer/e_agenda.h</a>	
Manage the agendas (timer2)	252
<a href="#">motor_led/advance_one_timer/e_led.c</a>	
Manage the LEDs with blinking possibility (timer2)	258
<a href="#">motor_led/advance_one_timer/e_led.h</a>	
Manage the LEDs with blinking possibility (timer2)	271
<a href="#">motor_led/advance_one_timer/e_motors.c</a>	
Manage the motors (with timer2)	285
<a href="#">motor_led/advance_one_timer/e_motors.h</a>	
Manage the motors (with timer2)	296
<a href="#">motor_led/advance_one_timer/e_remote_control.c</a>	
Manage the IR receiver module (timer2)	310
<a href="#">motor_led/advance_one_timer/e_remote_control.h</a>	
Manage the LEDs with blinking possibility (timer2)	312

<a href="#">motor_led/advance_one_timer/fast_agenda/e_agenda_fast.c</a>	
Manage the fast agendas (timer1, 2, 3) . . . . .	315
<a href="#">motor_led/advance_one_timer/fast_agenda/e_agenda_fast.h</a>	
Manage the fast agendas (timer1, 2, 3) . . . . .	319
<a href="#">motor_led/advance_one_timer/fast_agenda/e_led.c</a>	
Manage the LEDs with blinking possibility (timer1, 2, 3) . . . . .	264
<a href="#">motor_led/advance_one_timer/fast_agenda/e_led.h</a>	
Manage the LEDs with blinking possibility (timer1, 2, 3) . . . . .	278
<a href="#">motor_led/advance_one_timer/fast_agenda/e_motors.c</a>	
Manage the motors (with timer1, 2, 3) . . . . .	289
<a href="#">motor_led/advance_one_timer/fast_agenda/e_motors.h</a>	
Manage the motors (with timer1, 2, 3) . . . . .	301
<a href="#">uart/e_epuck_ports.inc</a> . . . . .	338
<a href="#">uart/e_uart_char.h</a>	
Manage UART . . . . .	338
<a href="#">utility/utility.c</a> . . . . .	341
<a href="#">utility/utility.h</a> . . . . .	343

# Chapter 5

## Module Documentation

### 5.1 Analogic/Digital conversion (ADC)

#### Files

- file [e\\_acc.c](#)  
*Accessing the accelerometer data.*
- file [e\\_acc.h](#)  
*Accessing the accelerometer data.*
- file [e\\_ad\\_conv.c](#)  
*Module for the advance Analogic/Digital conversion.*
- file [e\\_ad\\_conv.h](#)  
*Module for the advance Analogic/Digital conversion.*
- file [e\\_micro.c](#)  
*Accessing the microphone data.*
- file [e\\_micro.h](#)  
*Accessing the microphone data.*
- file [e\\_prox.c](#)  
*Accessing proximity sensor of e-puck.*
- file [e\\_prox.h](#)  
*Accessing proximity sensor of e-puck.*
- file [e\\_accelerometer.c](#)  
*Accessing the accelerometer sensor data.*
- file [e\\_accelerometer.h](#)  
*Accessing the accelerometer sensor data.*
- file [e\\_ad\\_conv.c](#)  
*Module for the Analogic/Digital conversion.*
- file [e\\_ad\\_conv.h](#)  
*Module for the Analogic/Digital conversion.*
- file [e\\_micro.c](#)  
*Accessing the microphone data.*
- file [e\\_micro.h](#)  
*Accessing the microphone data.*
- file [e\\_prox.c](#)  
*Accessing proximity sensor of e-puck with timer 1.*
- file [e\\_prox.h](#)  
*Accessing proximity sensor of e-puck with timer 1.*
- file [e\\_prox\\_timer2.c](#)  
*Control proximity sensor of e-puck with timer2.*

### 5.1.1 Detailed Description

### 5.1.2 Introduction

The microcontroller p30F6014A has a 12-bit Analog-to-Digital Converter (ADC). This package is built to manage this ADC.

The e-puck has three peripherals which take advantage from it.

- 1 3D accelerometer
- 8 proximity sensors
- 3 microphones

### 5.1.3 Package organization

This package is divided by two sub packages:

- The standard approach (files located in the "a\_d" folder). This approach uses the timer1 (or timer2) to coordinate the ADC register acquisition.
- The more advanced approach (files located in the "a\_d/advance\_ad\_scan" folder) uses the ADC interrupt to update the four **arrays** containing all the data of all the peripherals which use the ADC ([e\\_mic\\_scan](#) for the microphones, [e\\_acc\\_scan](#) for the accelerometer, [e\\_ambient\\_ir](#) and [e\\_ambient\\_and\\_reflected\\_ir](#) for the proximity sensors). In this approach, the acquisition is made automatically and always with the same delay. The functions specific to each module ([advance\\_ad\\_scan/e\\_acc.c](#), [advance\\_ad\\_scan/e\\_prox.c](#), [advance\\_ad\\_scan/e\\_micro.c](#)) are also more elaborates than the same one in the standard package.

#### Author

Doc: Jonathan Besuchet

## 5.2 Bluetooth

### Files

- file [e\\_bluetooth.c](#)  
*Manage Bluetooth.*
- file [e\\_bluetooth.h](#)  
*Manage Bluetooth.*

### 5.2.1 Detailed Description

### 5.2.2 Introduction

This package contains all the resources you need to control the bluetooth device you have on your e-puck AS MASTER. If you only want to use the bluetooth AS SLAVE, the uart library is enough (the connection to a master look like a standard uart connection).

The bluetooth device is connected on the uart1.

Generally, the bluetooth modules have a bluetooth class device which identify them as PC, mobile, mouse, ... The e-puck bluetooth module has the default device class number, it's 000.

To learn more about the e-puck's bluetooth device see the documentation of the LMX9820A from National Semiconductor (look at <http://www.national.com/mpf/LM/LMX9820A.html>) and look at the [UART](#) module.

### Author

Doc: Jonathan Besuchet

## 5.3 Camera fast two timers

### Files

- file [e\\_calc\\_po3030k.c](#)  
*Calculate the timing for the camera (two timers)*
- file [e\\_po3030k.h](#)  
*PO3030k library header (two timers)*
- file [e\\_po6030k.h](#)  
*PO6030k library header.*
- file [e\\_po8030d.h](#)  
*P08030d library header.*
- file [e\\_timers.c](#)  
*Manage camera's interrupts (two timers)*

### 5.3.1 Detailed Description

### 5.3.2 Introduction

This driver expose most of the Po3030k camera interfaces. Some functions are usefull, some other not. But since this is not up to the driver to decide if a function is needed, I've exported almost all.

The architecture is quite simple. The driver keep a array where every known camera register is kept in memory. The configuration function only alter this array. When you call, for example [e\\_po3030k\\_config\\_cam\(\)](#), nothing is written on the camera, but only in the internal register representation.

To effectively write the register, you must call [e\\_po3030k\\_write\\_cam\\_registers\(\)](#). This is typically done after every configuration call and before acquire the first picture.

### 5.3.3 Default settings

The camera is, by default, configured with the followin settings:

- Automatic white balance control
- Automatic exposure control
- Automatic flicker detection ( 50Hz and 60Hz )

There is no default setting for the image size and color.

### 5.3.4 Performances

The maximum framerate ( without doing anything else than acquiring the picture ) vary with the subsampling and the color mode. Here are some framerates:

- Size: 640x480, Subsampling: 16x16, RGB565: 4.3 fps
- Size: 16x480, Subsampling: 16x16, RGB565: 4.3 fps
- Size: 480x16, Subsampling: 16x16: RGB565: 4.3fps
- Size: 64x64, Subsampling: 4x4, RGB565: 4.3 fps
- Size: 32x32, Subsampling: 2x2, RGB565: 2.6 fps
- Size: 16x16, No Subsampling, RGB565: 1.3 fps
- Size: 640x480, Subsampling: 16x16, GREYSCALE: 8.6 fps
- Size: 16x480, Subsampling: 16x16, GREYSCALE: 8.6 fps
- Size: 480x16, Subsampling: 16x16, GREYSCALE: 8.6 fps
- Size: 64x64, Subsampling: 4x4, GREYSCALE: 8.6 fps
- Size: 32x32, Subsampling: 2x2, GREYSCALE: 4.3 fps
- Size: 16x16, No subsampling, GREYSCALE: 2.2 fps

### 5.3.5 Important note

This driver is extremely sensible to interrupt latency, thus it use interrupt priority to be sure that the latencies are kepts low. The Timer4 and Timer5 interrupt priority are set at level 6 and interrupt nesting is enabled. The Timer4 interrupt use the "push.s" and "pop.s" instructions. You should not have any code using thoses two instructions when you use the camera. This include the `_ISRFAST` C macro. If you use them, some random and really hard to debug behavior will happen. You have been warned !

### 5.3.6 Examples

#### 5.3.6.1 Basic example

```
#include "e_po3030k.h"

char buffer[2*40*40];
int main(void) {

    e_po3030k_init_cam();
    e_po3030k_config_cam((ARRAY_WIDTH -160)/2, (
ARRAY_HEIGHT-160)/2,
        160,160,4,4,RGB_565_MODE);
    e_po3030k_write_cam_registers();

    e_po3030k_launch_capture(buffer);
    while(!e_po3030k_is_img_ready());

    // buffer contain a 40*40 RGB picture now
    ( insert usefull code here )

    return 0;
}
```

This example tell de driver to aquire 160x160 pixel picture from the camera 4x subsampling, thus resulting with a 40x40 pixel. The buffer as a size of 40\*40\*2 because RGB565 is a two bytes per pixel data format.

### 5.3.6.2 More advanced example

```
#include "e_po3030k.h"

char buffer[160*2];
int main(void) {
    e_po3030k_init_cam();
    e_po3030k_config_cam((ARRAY_WIDTH - 320)/2, (
ARRAY_HEIGHT - 32)/2,
        320, 8, 2, 4, GREY_SCALE_MODE);
    e_po3030k_set_mirror(1, 1);
    e_po3030ke_set_ref_exposure(100);

    e_po3030k_write_cam_registers();

    e_po3030k_launch_capture(buffer);
    while(!e_po3030k_is_img_ready());

    // Here buffer contain a 160x2 greyscale picture

    return 0;
}
```

This example configure the camera to acquire a 320x8 pixel picture, but subsampled 2x in width and 4x in height, thus resulting in a 160\*2 linear greyscale picture. It "emulate" a linear camera. This example tell the camera to enable the vertical and horizontal mirror, and to set the average exposure to 100.

### 5.3.7 Introduction

This driver expose the subset features bewteen the po6030k and po3030k camera interfaces.



## 5.4 Camera slow three timers

### Files

- file [e\\_calc.c](#)  
*Calculate the timing for the camera (three timers)*
- file [e\\_po3030k.h](#)  
*PO3030k library header (three timers)*
- file [e\\_registers.c](#)  
*Manage po3030k registers (three timers)*
- file [e\\_timers.c](#)  
*Manage camera's interrupts (three timers)*

### 5.4.1 Detailed Description

#### Warning

This driver version is completely interrupt driven to synchronize with the camera. This slow down the acquisition a lot. You should use the other driver's version which synchronize with the camera only at each row.

### 5.4.2 Introduction

This driver expose most of the Po3030k camera interfaces. Some functions are usefull, some other not. But since this is not up to the driver to decide if a function is needed, I've exported almost all.

The architecture is quite simple. The driver keep a array where every known camera register is kept in memory. The configuration function only alter this array. When you call, for exemple [e\\_po3030k\\_config\\_cam\(\)](#), nothing is written on the camera, but only in the internal register representation.

To effectively write the register, you must call [e\\_po3030k\\_write\\_cam\\_registers\(\)](#). This is typically done after every configuration call and before acquire the first picture.

### 5.4.3 Default settings

The camera is, by default, configured with the followin settings:

- Automatic white balance control
- Automatic exposure control
- Automatic flicker detection ( 50Hz and 60Hz )

There is no default setting for the image size and color.

## 5.4.4 Performances

The maximum framerate ( without doing anything else than acquiring the picture ) vary with the subsampling and the color mode. Here are some framerates. Please use the other driver to have better performances :

- Size: 640x480, Subsampling: 16x16, RGB565: 0.6 fps
- Size: 64x64, Subsampling: 4x4, RGB565: 0.6 fps
- Size: 32x32, Subsampling: 2x2, RGB565: 0.3 fps
- Size: 16x16, No Subsampling, RGB565: 0.2 fps
- Size: 640x480, Subsampling: 16x16, GREYSCALE: 1.1 fps
- Size: 64x64, Subsampling: 4x4, GREYSCALE: 1.1 fps
- Size: 32x32, Subsampling: 2x2, GREYSCALE: 0.6 fps
- Size: 16x16, No subsampling, GREYSCALE: 0.3 fps

## 5.4.5 Exemples

### 5.4.5.1 Basic example

```
#include "e_po3030k.h"

char buffer[2*40*40];
int main(void) {

    e_po3030k_init_cam();
    e_po3030k_config_cam( (ARRAY_WIDTH -160)/2, (
ARRAY_HEIGHT-160)/2,
                        160,160,4,4,RGB_565_MODE);
    e_po3030k_write_cam_registers();

    e_po3030k_launch_capture(buffer);
    while(!e_po3030k_is_img_ready());

    // buffer contain a 40*40 RGB picture now
    ( insert usefull code here )

    return 0;
}
```

This exemple tell de driver to aquire 160x160 pixel picture from the camera 4x subsampling, thus resulting with a 40x40 pixel. The buffer as a size of 40\*40\*2 because RGB565 is a two bytes per pixel data format.

### 5.4.5.2 More advanced example

```
#include "e_po3030k.h"

char buffer[160*2];
int main(void) {
    e_po3030k_init_cam();
    e_po3030k_config_cam( (ARRAY_WIDTH - 320)/2, (
ARRAY_HEIGHT - 32)/2,
                        320,8,2,4,GREY_SCALE_MODE);
    e_po3030k_set_mirror(1,1);
    e_po3030k_set_ref_exposure(100);

    e_po3030k_write_cam_registers();

    e_po3030k_launch_capture(buffer);
    while(!e_po3030k_is_img_ready());

    // Here buffer contain a 160x2 greyscale picture

    return 0;
}
```

This exemple configure the camera to aquire a 320x8 pixel picture, but subsampled 2x in width and 4x in heighth, thus resulting in a 160\*2 linear greyscale picture. It "emulate" a linear camera. This exemple tell the camera to enable the vertical and horizontal mirror, and to set the average exposure to 100.

## 5.5 Sound

### Files

- file [e\\_sound.c](#)

*Package to play basics sounds on the e-puck's speaker.  
For more info look at this: [Sound](#).*

- file [e\\_sound.h](#)

*Package to play basics sounds on the e-puck's speaker.*

### 5.5.1 Detailed Description

### 5.5.2 Introduction

The e-puck has got a speaker on his top. This package is made to take advantage of it, by playing little sample.

### 5.5.3 Package organization

The internals functions of this package are written in assembler, because of speed. The sound you can play is in the file `codec/e_const_sound.s`

The externals functions are located in the file `codec/e_sound.c`. There are three functions: [e\\_init\\_sound\(void\)](#), [e\\_play\\_sound\(int sound\\_offset, int sound\\_length\)](#) and [void e\\_close\\_sound\(void\)](#). When you want to play a sound YOU HAVE TO call [e\\_init\\_sound\(void\)](#) at first, but only the first time.

To play a sound call [e\\_play\\_sound\(int sound\\_offset, int sound\\_length\)](#). This function takes two parameters, the first set the beginning of the sound, the second set the length of the sound. In fact it works like this:

- The "sound" which is in the file `codec/e_const_sound.s` is placed somewhere in e-puck's memory.
- When you call the function [e\\_play\\_sound\(int sound\\_offset, int sound\\_length\)](#), the words are sent to the DCI one by one from the offset you have specified with the first parameter. The number of words sent are specified with the second parameter.

If you don't want to use the sound anymore call [e\\_close\\_sound](#).

#### Warning

If you call [void e\\_close\\_sound\(void\)](#), you have to recall [ee\\_init\\_sound\(void\)](#) to play sound again.

## 5.5.4 Sounds plage

In the file `codec/e_const_sound.s` there are 19044 words. Five sounds are organized as following [begining, length]:

- [0, 2112]: "haa"
- [2116, 1760]: "spaah"
- [3878, 3412]: "ouah"
- [7294, 3732]: "yaouh"
- [11028, 8016]: "wouaaaaaaaah"

Then if you want to play the "yaouh" sound from the begining to the end just write this: `e_play_sound(7294, 3732);`

A little exemple which plays the five sounds one by one.

```
#include <codec/e_sound.h>
#include <motor_led/e_init_port.h>

int main(void)
{
    e_init_port();
    e_init_sound();
    while(1)
    {
        long i;
        e_play_sound(0, 2112); //sound 1
        for(i=0; i<4000000; i++) {asm("nop");}
        e_play_sound(2116, 1760); //sound 2
        for(i=0; i<4000000; i++) {asm("nop");}
        e_play_sound(3878, 3412); //sound 3
        for(i=0; i<4000000; i++) {asm("nop");}
        e_play_sound(7294, 3732); //sound 4
        for(i=0; i<4000000; i++) {asm("nop");}
        e_play_sound(11028, 8016); //sound 5
        for(i=0; i<4000000; i++) {asm("nop");}
    }
}
```

### Author

Code: Michael Bonani

Doc: Jonathan Besuchet

## 5.6 LIS sensor turret

### Files

- file [e\\_devantech.h](#)  
*Devantech sensor of e-puck.*

### 5.6.1 Detailed Description

### 5.6.2 Introduction

This module is made for an extension of the e-puck. No documentation is available for now.

## 5.7 Radio communication

### Files

- file [ComModule.h](#)  
*Radio communication.*

### 5.7.1 Detailed Description

### 5.7.2 Introduction

This module is made for an extension of the e-puck. No documentation is available for now.

## 5.8 FFT

### Files

- file [e\\_fft.c](#)  
*Package to manage the FFT.*
- file [e\\_fft.h](#)  
*Package to manage the FFT.*
- file [e\\_fft\\_utilities.h](#)  
*Some fft features.*
- file [e\\_input\\_signal.c](#)  
*Allocate memory and initialize the sigCmpx array.*
- file [e\\_twiddle\\_factors.c](#)  
*The FFT factor from Microchip.*

### 5.8.1 Detailed Description

### 5.8.2 Introduction

The fast fourier transform (FFT) is really usefull, especially to work with the microphones data. This package contains all you need to perform the FFT.

The dsPic has some specials instructions (MAC instructions) which are used here.

### 5.8.3 How it works

To do the FFT on your data, first make this:

- choose the size of the array in which the FFT will be done. You have to choose one of the following values: 64, 128, 256 or 512.
- put your choice in the [FFT\\_BLOCK\\_LENGTH](#) (it's in the file [e\\_fft.h](#)).

Then you just have

- to copy your data in the sigCmpx array with the [e\\_fast\\_copy\(int\\* in\\_array, int\\* out\\_array, int size\)](#) function
- to call the [e\\_doFFT\\_asm\(fractcomplex\\* sigCmpx\)](#) function

A little code to illustrate this.

```
e_ad_scan_on();
// waiting all the 512 data (here we scan the microphones)
while(!e_ad_is_array_filled());
e_ad_scan_off();

// We put the mean to zero
e_subtract_mean(e_mic_scan[0], FFT_BLOCK_LENGTH,
LOG2_BLOCK_LENGTH);
// We copy the array of micro zero to the FFT array
e_fast_copy(e_mic_scan[0], (int*)sigCmpx,
FFT_BLOCK_LENGTH);

// The result is saved => we can launch a new acquisition
e_ad_scan_on();
// Now we are doing the FFT
e_doFFT_asm(sigCmpx);
```

### Author

Doc: Jonathan Besuchet

## 5.9 I2C

### Files

- file [e\\_i2c\\_master\\_module.c](#)  
*Manage I2C basics.*
- file [e\\_i2c\\_master\\_module.h](#)  
*Manage I2C basics.*
- file [e\\_i2c\\_protocol.c](#)  
*Manage I2C protocole.*
- file [e\\_i2c\\_protocol.h](#)  
*Manage I2C protocole.*

### 5.9.1 Detailed Description

### 5.9.2 Introduction

This software allows using the I2C hardware module on a DsPic30f60xx in a master mode for a single master system.

This module manage the I2C basics functions (low level I2C functions). They are made to perform the basics tasks like:

- initializing the I2C on the microcontroller ([e\\_i2c\\_init\(void\)](#))
- sending the Start bit ([e\\_i2c\\_start\(void\)](#))
- sending the Restart bit ([e\\_i2c\\_restart\(void\)](#))
- sending the Stop bit ([e\\_i2c\\_stop\(void\)](#))
- sending the acknowledgement bit ([e\\_i2c\\_ack\(void\)](#))
- writing or receiving a byte ([e\\_i2c\\_write\(char byte\)](#), [e\\_i2c\\_read\(char \\*buf\)](#))
- ...

### 5.9.3 Overview of I2C protocol

The I2C-bus supports any IC fabrication process (NMOS, CMOS, bipolar). Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. Each device is recognized by a unique address (whether it's a microcontroller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the device.



### 5.9.3.1 Master-slave relation

The I2C-bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. As masters are usually microcontrollers, let's consider the case of a data transfer between two microcontrollers connected to the I2C-bus.

This highlights the master-slave and receiver-transmitter relationships to be found on the I2C-bus. It should be noted that these relationships are not permanent, but only depend on the direction of data transfer at that time. The transfer of data would proceed as follows:

1) Suppose microcontroller A wants to send information to module B:

- microcontroller A (master), addresses module B (slave)
- microcontroller A (master-transmitter), sends data to module B (slave-receiver)
- microcontroller A terminates the transfer

2) If microcontroller A wants to receive information from module B:

- microcontroller A (master) addresses module B (slave)
- microcontroller A (master-receiver) receives data from module B (slave-transmitter)
- microcontroller A terminates the transfer.

Even in this case, the master (microcontroller A) generates the timing and terminates the transfer.

### 5.9.3.2 Start and Stop conditions

Within the procedure of the I2C-bus, unique situations arise which are defined as START (S) and STOP (P) conditions.

A HIGH to LOW transition on the SDA line while SCL is HIGH is one such unique case. This situation indicates a START condition.

A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition.

START and STOP conditions are always generated by the master. The bus is considered to be busy after the START condition. The bus is considered to be free again a certain time after the STOP condition.

The bus stays busy if a repeated START (Sr) is generated instead of a STOP condition. In this respect, the START (S) and repeated START (Sr) conditions are functionally identical). Detection of START and STOP conditions by devices connected to the bus is easy if they incorporate the necessary interfacing hardware.

### 5.9.3.3 Transferring data

Every byte put on the SDA line must be 8-bits long (char type).

Acknowledge:

Data transfer with acknowledge is obligatory. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (HIGH) during the acknowledge clock pulse.

The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse. Of course, set-up and hold times must also be taken into account.

Usually, a receiver which has been addressed is obliged to generate an acknowledge after each byte has been received.

When a slave doesn't acknowledge the slave address (for example, it's unable to receive or transmit because it's performing some real-time function), the data line must be left HIGH by the slave. The master can then generate either a STOP condition to abort the transfer, or a repeated START condition to start a new transfer. If a slave-receiver does acknowledge the slave address but, some time later in the transfer cannot receive any more data bytes, the master must again abort the transfer. This is indicated by the slave generating the not-acknowledge on the first byte to follow. The slave leaves the data line HIGH and the master generates a STOP or a repeated START condition.

If a master-receiver is involved in a transfer, it must signal the end of data to the slave-transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave-transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

## 5.9.4 Use I2C on e-puck

On the e-puck, the microcontroller is always the master.

### 5.9.4.1 Basics functions

The functions of the files [e\\_i2c\\_master\\_module.c](#) and [e\\_i2c\\_master\\_module.h](#) are low level I2C functions. They are made to perform the basics tasks like:

- initializing the I2C on the microcontroller ([e\\_i2c\\_init\(void\)](#))
- sending the Start bit ([e\\_i2c\\_start\(void\)](#))
- sending the Restart bit ([e\\_i2c\\_restart\(void\)](#))
- sending the Stop bit ([e\\_i2c\\_stop\(void\)](#))
- sending the acknowledgement bit ([e\\_i2c\\_ack\(void\)](#))
- writing or receiving a byte ([e\\_i2c\\_write\(char byte\)](#), [e\\_i2c\\_read\(char \\*buf\)](#))
- ...

### 5.9.4.2 More developed functions

The functions of the files [e\\_i2c\\_protocol.c](#) and [e\\_i2c\\_protocol.h](#) are made to directly send or receive data from or to a specified slave.

## 5.9.5 Reference

For more information about I2C:

- <http://en.wikipedia.org/wiki/I%C2%B2C>
- [http://www.nxp.com/acrobat\\_download/literature/9398/39340011.pdf](http://www.nxp.com/acrobat_download/literature/9398/39340011.pdf)
- <http://tmd.havit.cz/Papers/I2C.pdf>

### Author

Doc: Jonathan Besuchet

## 5.10 Matlab communication

### Files

- file [matlab.c](#)  
*To communicate with matlab.*
- file [matlab.h](#)  
*To communicate with matlab.*

### 5.10.1 Detailed Description

### 5.10.2 Introduction

This package contains all the resources you need to communicate with matlab through bluetooth.

To make the communication possible, you have to follow this steps:

- Open matlab and set the default directories to "...\\library\\matlab\\matlab files"
- Connect your e-puck to your PC with bluetooth
- Call the matlab function "OpenEpuck("COMX"); X is the number of the port on which the e-puck is connected.

#### 5.10.2.1 Sending data from matlab

If you want to send data from matlab you only have to call the matlab function "EpuckSendData(data, dataType)"

- data is the array of value you want to send;
- dataType is a string argument which can be 'int8' to send char or 'int16' to send int;

On the e-puck side, to receive the data, you have to call the appropriate function depending of the data type you will receive. For exemple if matlab send int data, you have to call: [e\\_receive\\_int\\_from\\_matlab\(int \\*data, int array\\_size\)](#).

#### 5.10.2.2 Sending data from e-puck

Now if you want to send data from e-puck to matlab you have to call the function [e\\_send\\_int\\_to\\_matlab\(int\\* data, int array\\_size\)](#) (send int data) on e-puck side and call "EpuckGetData" on matlab side. This function make the data conversion automatically, so call it to receive both of 'char' or 'int' data.

### See also

[Matlab communication](#)

### Author

Doc: Jonathan Besuchet

## 5.11 Ports, motors and LEDs

### Files

- file [e\\_agenda.c](#)  
*Manage the agendas (timer2)*
- file [e\\_agenda.h](#)  
*Manage the agendas (timer2)*
- file [e\\_led.c](#)  
*Manage the LEDs with blinking possibility (timer2).*
- file [e\\_led.h](#)  
*Manage the LEDs with blinking possibility (timer2).*
- file [e\\_motors.c](#)  
*Manage the motors (with timer2)*
- file [e\\_motors.h](#)  
*Manage the motors (with timer2)*
- file [e\\_remote\\_control.c](#)  
*Manage the IR receiver module (timer2)*
- file [e\\_remote\\_control.h](#)  
*Manage the LEDs with blinking possibility (timer2).*
- file [e\\_remote\\_control.h](#)  
*Manage the LEDs with blinking possibility (timer2).*
- file [e\\_agenda\\_fast.c](#)  
*Manage the fast agendas (timer1, 2, 3).*
- file [e\\_agenda\\_fast.h](#)  
*Manage the fast agendas (timer1, 2, 3).*
- file [e\\_led.c](#)  
*Manage the LEDs with blinking possibility (timer1, 2, 3).*
- file [e\\_led.h](#)  
*Manage the LEDs with blinking possibility (timer1, 2, 3).*
- file [e\\_motors.c](#)  
*Manage the motors (with timer1, 2, 3)*
- file [e\\_motors.h](#)  
*Manage the motors (with timer1, 2, 3)*
- file [e\\_epuck\\_ports.h](#)  
*Define all the usefull names corresponding of e-puck's hardware.*
- file [e\\_init\\_port.c](#)  
*Initialize the ports on standard configuration.*
- file [e\\_init\\_port.h](#)  
*Initialize the ports on standard configuration.*
- file [e\\_led.c](#)  
*Manage the LEDs.  
A little exemple for the LEDs (all the LEDs are blinking)*
- file [e\\_led.h](#)  
*Manage the LEDs.  
A little exemple for the LEDs (all the LEDs are blinking)*
- file [e\\_motors.c](#)  
*Manage the motors (with timer 4 and 5).*
- file [e\\_motors.h](#)  
*Manage the motors (with timer 4 and 5).*
- file [e\\_motors\\_timer3.c](#)  
*Initialize the ports on standard configuration.*
- file [e\\_motors\\_timer3.c](#)  
*Initialize the ports on standard configuration.*

## 5.11.1 Detailed Description

### 5.11.2 Introduction

This package contains all the resources you need to control the ports, the motors, the LED and the IR receiver of the e-puck.

#### 5.11.2.1 Ports

The standard port's name of the p30F6014A microcontroller is not explicit in the e-puck context, so we need to redefine these names to make them more user friendly.

This work is made in the file: [e\\_epuck\\_ports.h](#).

#### 5.11.2.2 Motors

The e-puck has two step by step motors called MOTOR1 (left) and MOTOR2 (right). To control the changing phase's sequence of these motors we need to use timers. Four possibilities are offered to you:

- standard: we use the timer4 for MOTOR2 and timer5 for MOTOR1. This solution is exploited by the file [library\motor\\_led\e\\_motors.c](#).
- one timer standard: we use the timer3 for both MOTOR1 and MOTOR2. This solution is exploited by the file [library\motor\\_led\e\\_motors\\_timer3.c](#)
- advance one timer: we use the timer2 for both MOTOR1 and MOTOR2, but this time the mechanism work on the agenda method (see below or [e\\_agenda.h](#) for more information about agenda). This solution is exploited by the file [library\motor\\_led\advance\\_one\\_timer\e\\_motors.c](#).
- fast agenda: we use the timer1,2,3 for both MOTOR1 and MOTOR2, but this time the mechanism work on the fast\_agenda method (see below or [e\\_agenda\\_fast.h](#) for more information about fast\_agenda). This solution is exploited by the file [library\motor\\_led\advance\\_one\\_timer/fast\\_agenda\e\\_motors.c](#).

#### 5.11.2.3 LED

The e-puck has 8 reds LEDs, a front LED and a body LED. All the functions needed to control these LEDs are in the file [library\motor\\_led\e\\_led.c](#). This file is made for basics use. If you want blinking functions you have to work with these following files: [library\motor\\_led\advance\\_one\\_timer\e\\_led.c](#) or [library\motor\\_led\advance\\_one\\_timer/fast\\_agenda\e\\_led.c](#). In the case you will work with agenda solution (see below or [e\\_agenda.h](#) or [e\\_agenda\\_fast.h](#) for more information about agenda or fast agenda).

#### 5.11.2.4 IR remote

The e-puck has a IR receptor. To control this receptor look at this file: [library\motor\\_led\advance\\_one\\_timer\e\\_remote\\_control.c](#).

#### Warning

The IR remote uses the agenda solution, then it use timer2 (see below or [e\\_agenda.h](#) for more information about agenda).

### 5.11.3 Timer's problems

The p30F6014A microcontroller has five timers. The camera's package uses the timer4 and the timer5, so we can't exploit them to make the motors work when we want to use the camera. For this reason we can't use the standard solution above.

#### Warning

If you are using the camera, you have to work with one of this three solutions explained above:

- one timer standard
- advance one timer
- fast agenda

### 5.11.4 Agenda solution

As we have seen, we can use the agenda solution to make the motors work.

So what is an agenda ?

An agenda is a structure which can launch a specific function (called callback function) with a given intervals. The agenda structure is made to work as chained list.

How it works ?

You create an agenda by specifying:

- the callback function you will call
- the delay between two calls
- the next element of the chained list

On each timer overflow all the agenda chained list is scanned and if an agenda in the list has reached the delay, then the callback function is called.

See also

[e\\_agenda.c](#), [e\\_agenda\\_fast.c](#)

Author

Doc: Jonathan Besuchet

## 5.12 UART

### Files

- file [e\\_uart\\_char.h](#)  
*Manage UART.*

### 5.12.1 Detailed Description

### 5.12.2 Introduction

This package contains all the resources you need to control the UART (universal asynchronous receiver transmitter). The microcontroller p30f6014A has two UART controller: UART1 and UART2.

#### Warning

In this package, the functions are written in ASM. We have "e\_init\_uartX.s" file for the initializing functions, "e\_uartX\_rx.s" file for receiving data functions and "e\_uartX\_tx.s" file for transmitting data functions (X can be 1 or 2).

Even these files are written in ASM, you can call them by including the [e\\_uart\\_char.h](#) files in your C code (look at the exemple in [e\\_uart\\_char.h](#)).

### 5.12.3 Bluetooth

The e-puck has his bluetooth module connected on the uart1. Two ways are possibles when you work with bluetooth:

- you are the master, look at this: [Bluetooth](#)
- you are the slave.

When you are the slave, you can communicate with the master device exactly by the same way as you do to communicate through the uart. The bluetooth protocole is made to look like as transparent as possible. This is possible because the master initialize the communication and the bluetooth module answer automatically to create the connection. After that the connection was created, you can communicate with the master by using the uart protocole.

#### Author

Doc: Jonathan Besuchet





# Chapter 6

## Data Structure Documentation

### 6.1 AgendaList Struct Reference

Manage the different agendas lists.

```
#include <e_agenda_fast.h>
```

#### Data Fields

- char `motors`
- Agenda \* `waiting`
- Agenda \* `agendas` [3]
- unsigned char `timers_in_use` [3]
- unsigned `speed` [3]

#### 6.1.1 Detailed Description

Manage the different agendas lists.

We use the 3 first timers, then we need 3 pointers of the beginning of each Agenda chained list. We also have a pointer of the waiting Agenda chained list.

#### 6.1.2 Field Documentation

##### 6.1.2.1 Agenda\* AgendaList::agendas[3]

We use the 3 first timers

##### 6.1.2.2 char AgendaList::motors

##### 6.1.2.3 unsigned AgendaList::speed[3]

Base speed 0.1 ms but use of multiplication

#### 6.1.2.4 unsigned char AgendaList::timers\_in\_use[3]

Determine which one we use currently

#### 6.1.2.5 Agenda\* AgendaList::waiting

If an agenda's speed goes down to zero, we remove it from the list and add it to the waiting list

The documentation for this struct was generated from the following file:

- [motor\\_led/advance\\_one\\_timer/fast\\_agenda/e\\_agenda\\_fast.h](#)

## 6.2 AgendaType Struct Reference

struct Agenda as chained list

```
#include <e_agenda.h>
```

### Data Fields

- unsigned int [cycle](#)
- int [counter](#)
- char [activate](#)
- void(\* [function](#) )(void)
- [Agenda](#) \* [next](#)

### 6.2.1 Detailed Description

struct Agenda as chained list

The role of Agenda is to launch the pointed function when the member "counter" is greater than the member "cycle". The member "activate" can be on=1 or off=0. When it is off, the counter don't increase.

This struct is designed to be used as chained list so we need a pointer to the next element.

### 6.2.2 Field Documentation

#### 6.2.2.1 char AgendaType::activate

can be on=1 or off=0

#### 6.2.2.2 int AgendaType::counter

count the number of interrupts

### 6.2.2.3 unsigned int AgendaType::cycle

length in 10e of ms of a cycle between two events

### 6.2.2.4 void(\* AgendaType::function)(void)

function called when counter > cycle,

#### Warning

This function must have the following prototype: "void func(void)"

### 6.2.2.5 Agenda \* AgendaType::next

pointer on the next agenda

The documentation for this struct was generated from the following files:

- [motor\\_led/advance\\_one\\_timer/e\\_agenda.h](#)
- [motor\\_led/advance\\_one\\_timer/fast\\_agenda/e\\_agenda\\_fast.h](#)

## 6.3 BtDevice Struct Reference

general struct for bluetooth device

```
#include <e_bluetooth.h>
```

### Data Fields

- unsigned char [address](#) [6]
- unsigned char [class](#) [3]
- char [friendly\\_name](#) [20]

### 6.3.1 Detailed Description

general struct for bluetooth device

### 6.3.2 Field Documentation

#### 6.3.2.1 unsigned char BtDevice::address[6]

Bluetooth MAC address

### 6.3.2.2 unsigned char BtDevice::class[3]

Bluetooth class of device

### 6.3.2.3 char BtDevice::friendly\_name[20]

The friendly name of the bluetooth device

The documentation for this struct was generated from the following file:

- [bluetooth/e\\_bluetooth.h](#)

## 6.4 BtEPuck Struct Reference

general struct for other e-puck

```
#include <e_bluetooth.h>
```

### Data Fields

- unsigned char [address](#) [6]
- unsigned char [number](#) [5]

### 6.4.1 Detailed Description

general struct for other e-puck

### 6.4.2 Field Documentation

#### 6.4.2.1 unsigned char BtEPuck::address[6]

e-puck's bluetooth MAC address

#### 6.4.2.2 unsigned char BtEPuck::number[5]

e-puck's bluetooth PIN

The documentation for this struct was generated from the following file:

- [bluetooth/e\\_bluetooth.h](#)

## 6.5 TypeAccRaw Struct Reference

struct to store the acceleration raw data in cartesian coord

```
#include <e_acc.h>
```

### Data Fields

- int [acc\\_x](#)
- int [acc\\_y](#)
- int [acc\\_z](#)

### 6.5.1 Detailed Description

struct to store the acceleration raw data in cartesian coord

### 6.5.2 Field Documentation

#### 6.5.2.1 int TypeAccRaw::acc\_x

The acceleration on x axis

#### 6.5.2.2 int TypeAccRaw::acc\_y

The acceleration on y axis

#### 6.5.2.3 int TypeAccRaw::acc\_z

The acceleration on z axis

The documentation for this struct was generated from the following file:

- [a\\_d/advance\\_ad\\_scan/e\\_acc.h](#)

## 6.6 TypeAccSpheric Struct Reference

struct to store the acceleration vector in spherical coord

```
#include <e_acc.h>
```

### Data Fields

- float [acceleration](#)
- float [orientation](#)
- float [inclination](#)

### 6.6.1 Detailed Description

struct to store the acceleration vector in spherical coord

### 6.6.2 Field Documentation

#### 6.6.2.1 float TypeAccSpheric::acceleration

length of the acceleration vector = intensity of the acceleration

#### 6.6.2.2 float TypeAccSpheric::inclination

inclination angle with the horizontal plan

- 0° = e-puck horizontal
- 90° = e-puck vertical
- 180° = e-puck horizontal but up-side-down

#### 6.6.2.3 float TypeAccSpheric::orientation

orientation of the acceleration vector in the horizontal plan, zero facing front

- 0° = inclination to the front (front part lower than rear part)
- 90° = inclination to the left (left part lower than right part)
- 180° = inclination to the rear (rear part lower than front part)
- 270° = inclination to the right (right part lower than left part)

The documentation for this struct was generated from the following file:

- [a\\_d/advance\\_ad\\_scan/e\\_acc.h](#)

# Chapter 7

## File Documentation

### 7.1 a\_d/advance\_ad\_scan/e\_acc.c File Reference

Accessing the accelerometer data.

```
#include "math.h"
#include "e_acc.h"
#include "e_ad_conv.h"
#include "../motor_led/e_epuck_ports.h"
#include "../motor_led/advance_one_timer/e_led.h"
#include "../acc_gyro/e_lsm330.h"
#include "../motor_led/e_init_port.h"
#include <stdlib.h>
```

#### Functions

- int [e\\_get\\_acc](#) (unsigned int captor)  
*Read the last value of a given accelerator axes.*
- int [e\\_get\\_acc\\_filtered](#) (unsigned int captor, unsigned int filter\_size)  
*Read the value of a channel, filtered by an averaging filter.*
- void [calculate\\_acc\\_raw](#) (void)  
*read the x, y, z values, apply an averaging filter and finally subtract the center values.*
- void [calculate\\_acc\\_spherical](#) (void)  
*calculate the intensity of the acceleration vector, and the Euler's angles*
- void [e\\_acc\\_calibr](#) (void)  
*initialize the ad converter and calculate the zero values*
- [TypeAccSpheric e\\_read\\_acc\\_spheric](#) (void)  
*calculate and return the accel. in spherical coord*
- float [e\\_read\\_inclination](#) (void)  
*calculate and return the inclination angle*
- float [e\\_read\\_orientation](#) (void)  
*calculate and return the orientation angle*
- float [e\\_read\\_acc](#) (void)  
*calculate and return the intensity of the acceleration*
- [TypeAccRaw e\\_read\\_acc\\_xyz](#) (void)

- calculate and return acceleration on the x,y,z axis*
- int [e\\_read\\_acc\\_x](#) (void)
  - calculate and return acceleration on the x axis*
- int [e\\_read\\_acc\\_y](#) (void)
  - calculate and return acceleration on the y axis*
- int [e\\_read\\_acc\\_z](#) (void)
  - calculate and return acceleration on the z axis*
- void [e\\_display\\_angle](#) (void)
  - light the led according to the orientation angle*

## Variables

- unsigned char [updateAccI2CCounter](#)
- int [e\\_acc\\_scan](#) [3][[ACC\\_SAMP\\_NB](#)]
- unsigned int [e\\_last\\_acc\\_scan\\_id](#)
- unsigned char [isCalibratingFlag](#) = 0
- static int [angle\\_mem](#) = 0
- static int [centre\\_x](#) = 0
- static int [centre\\_y](#) = 0
- int [centre\\_z](#) = 2000
- static int [acc\\_x](#)
- static int [acc\\_y](#)
- static int [acc\\_z](#)
- static float [acceleration](#)
- static float [orientation](#)
- static float [inclination](#)
- int [lastAccX](#)
- int [lastAccY](#)
- int [lastAccZ](#)

### 7.1.1 Detailed Description

Accessing the accelerometer data.

The functions of this file are made to deal with the accelerometer data. You can know the magnitude, the orientation, the inclination, ... of the acceleration that the e-puck is enduring.

Two structures are used:

- [TypeAccSpheric](#) to store the acceleration data on spherical coordinates.
- [TypeAccRaw](#) to store the acceleration data on cartesian coordinates.

A little exemple to read the accelerator.



```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/advance_ad_scan/e_ad_conv.h>
#include <a_d/advance_ad_scan/e_acc.h>

int main(void)
{
    int z;
    e_init_port();
    e_init_ad_scan();
    while(1)
    {
        long i;
        z = e_get_acc(2);
        if(z < 2100) //LED4 on if e-puck is on the back
        {
            LED0 = 0;
            LED4 = 1;
        }
        else //LED0 on if e-puck is on his wells
        {
            LED0 = 1;
            LED4 = 0;
        }
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

## Author

Code: Borter Jean-Joel  
Doc: Jonathan Besuchet

## 7.1.2 Function Documentation

### 7.1.2.1 void calculate\_acc\_raw ( void )

read the x, y, z values, apply an averaging filter and finally substract the center values.

### 7.1.2.2 void calculate\_acc\_spherical ( void )

calculate the intensity of the acceleration vector, and the Euler's angles

### 7.1.2.3 void e\_acc\_calibr ( void )

initialize de ad converter and calculate the zero values

It reads two times the average\_size to avoid edge effects then it reads 100 values and average them to initiate the "zero" value of the accelerometer

### 7.1.2.4 void e\_display\_angle ( void )

light the led according to the orientation angle

### 7.1.2.5 int e\_get\_acc ( unsigned int *captor* )

Read the last value of a given accelerator axes.

## Parameters

<i>captor</i>	ID of the AD channel to read (must be 0 = x, 1 = y or 2 = z)
---------------	--

## Returns

value filtered channel's value

7.1.2.6 int e\_get\_acc\_filtered ( unsigned int *captor*, unsigned int *filter\_size* )

Read the value of a channel, filtered by an averaging filter.

## Parameters

<i>captor</i>	ID of the AD channel to read (must be 0 to 2)
<i>filter_size</i>	size of the filter (must be between 1 and SAMPLE_NUMBER)

## Returns

value filtered channel's value

## 7.1.2.7 float e\_read\_acc ( void )

calculate and return the intensity of the acceleration

## Returns

intensity of the acceleration vector

## 7.1.2.8 TypeAccSpheric e\_read\_acc\_spheric ( void )

calculate and return the accel. in spherical coord

## Returns

acceleration in spherical coord

## See also

[TypeAccSpheric](#)

## 7.1.2.9 int e\_read\_acc\_x ( void )

calculate and return acceleration on the x axis

## Returns

acceleration on the x axis

**7.1.2.10** `TypeAccRaw e_read_acc_xyz( void )`

calculate and return acceleration on the x,y,z axis

**Returns**

acceleration on the x,y,z axis

**See also**

[TypeAccRaw](#)

**7.1.2.11** `int e_read_acc_y( void )`

calculate and return acceleration on the y axis

**Returns**

acceleration on the y axis

**7.1.2.12** `int e_read_acc_z( void )`

calculate and return acceleration on the z axis

**Returns**

acceleration on the z axis

**7.1.2.13** `float e_read_inclination( void )`

calculate and return the inclination angle

**Returns**

inclination angle of the robot

**7.1.2.14** `float e_read_orientation( void )`

calculate and return the orientation angle

**Returns**

orientation of the accel vector

### 7.1.3 Variable Documentation

7.1.3.1 `int acc_x` [static]

7.1.3.2 `int acc_y` [static]

7.1.3.3 `int acc_z` [static]

7.1.3.4 `float acceleration` [static]

7.1.3.5 `int angle_mem = 0` [static]

7.1.3.6 `int centre_x = 0` [static]

7.1.3.7 `int centre_y = 0` [static]

7.1.3.8 `int centre_z = 2000`

7.1.3.9 `int e_acc_scan[3][ACC_SAMP_NB]`

Array to store the acc values

7.1.3.10 `unsigned int e_last_acc_scan_id`

7.1.3.11 `float inclination` [static]

7.1.3.12 `unsigned char isCalibratingFlag = 0`

7.1.3.13 `int lastAccX`

7.1.3.14 `int lastAccY`

7.1.3.15 `int lastAccZ`

7.1.3.16 `float orientation` [static]

7.1.3.17 `unsigned char updateAccI2CCounter`

## 7.2 `a_d/advance_ad_scan/e_acc.h` File Reference

Accessing the accelerometer data.

## Data Structures

- struct [TypeAccSpheric](#)  
*struct to store the acceleration vector in spherical coord*
- struct [TypeAccRaw](#)  
*struct to store the acceleration raw data in carthesian coord*

## Macros

- #define [CST\\_RADIAN](#) (180.0/3.1415)
- #define [ANGLE\\_ERROR](#) 666.0
- #define [FILTER\\_SIZE](#) 5
- #define [ACCX\\_BUFFER](#) 0
- #define [ACCY\\_BUFFER](#) 1
- #define [ACCZ\\_BUFFER](#) 2
- #define [GRAVITY](#) 768
- #define [GRAVITY\\_LSM330](#) 16384

## Functions

- int [e\\_get\\_acc](#) (unsigned int captor)  
*Read the last value of a given accelerator axes.*
- int [e\\_get\\_acc\\_filtered](#) (unsigned int captor, unsigned int filter\_size)  
*Read the value of a channel, filtered by an averaging filter.*
- [TypeAccSpheric e\\_read\\_acc\\_spheric](#) (void)  
*calculate and return the accel. in spherical coord*
- float [e\\_read\\_orientation](#) (void)  
*calculate and return the orientation angle*
- float [e\\_read\\_inclination](#) (void)  
*calculate and return the inclination angle*
- float [e\\_read\\_acc](#) (void)  
*calculate and return the intensity of the acceleration*
- [TypeAccRaw e\\_read\\_acc\\_xyz](#) (void)  
*calculate and return acceleration on the x,y,z axis*
- int [e\\_read\\_acc\\_x](#) (void)  
*calculate and return acceleration on the x axis*
- int [e\\_read\\_acc\\_y](#) (void)  
*calculate and return acceleration on the y axis*
- int [e\\_read\\_acc\\_z](#) (void)  
*calculate and return acceleration on the z axis*
- void [e\\_acc\\_calibr](#) (void)  
*initialize de ad converter and calculate the zero values*
- void [e\\_display\\_angle](#) (void)  
*light the led according to the orientation angle*

## 7.2.1 Detailed Description

Accessing the accelerometer data.

The functions of this file are made to deal with the accelerometer data. You can know the magnitude, the orientation, the inclination, ... of the acceleration that the e-puck is enduring.

Two structures are used:

- [TypeAccSpheric](#) to store the acceleration data on sherical coordinates.
- [TypeAccRaw](#) to store the acceleration data on cartesian coordinates.

A little exemple to read the accelerator.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/advance_ad_scan/e_ad_conv.h>
#include <a_d/advance_ad_scan/e_acc.h>

int main(void)
{
    int z;
    e_init_port();
    e_init_ad_scan();
    while(1)
    {
        long i;
        z = e_get_acc(2);
        if(z < 2100) //LED4 on if e-puck is on the back
        {
            LED0 = 0;
            LED4 = 1;
        }
        else //LED0 on if e-puck is on his wells
        {
            LED0 = 1;
            LED4 = 0;
        }
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

### Author

Code: Borter Jean-Joël  
Doc: Jonathan Besuchet

## 7.2.2 Macro Definition Documentation

### 7.2.2.1 #define ACCX\_BUFFER 0

### 7.2.2.2 #define ACCY\_BUFFER 1

### 7.2.2.3 #define ACCZ\_BUFFER 2

### 7.2.2.4 #define ANGLE\_ERROR 666.0

### 7.2.2.5 #define CST\_RADIAN (180.0/3.1415)

7.2.2.6 #define FILTER\_SIZE 5

7.2.2.7 #define GRAVITY 768

7.2.2.8 #define GRAVITY\_LSM330 16384

## 7.2.3 Function Documentation

7.2.3.1 void e\_acc\_calibr ( void )

initialize the ad converter and calculate the zero values

It reads two times the average\_size to avoid edge effects then it reads 100 values and average them to initiate the "zero" value of the accelerometer

7.2.3.2 void e\_display\_angle ( void )

light the led according to the orientation angle

7.2.3.3 int e\_get\_acc ( unsigned int *captor* )

Read the last value of a given accelerator axes.

### Parameters

<i>captor</i>	ID of the AD channel to read (must be 0 = x, 1 = y or 2 = z)
---------------	--

### Returns

value filtered channel's value

7.2.3.4 int e\_get\_acc\_filtered ( unsigned int *captor*, unsigned int *filter\_size* )

Read the value of a channel, filtered by an averaging filter.

### Parameters

<i>captor</i>	ID of the AD channel to read (must be 0 to 2)
<i>filter_size</i>	size of the filter (must be between 1 and SAMPLE_NUMBER)

### Returns

value filtered channel's value

#### 7.2.3.5 float e\_read\_acc ( void )

calculate and return the intensity of the acceleration

##### Returns

intensity of the acceleration vector

#### 7.2.3.6 TypeAccSpheric e\_read\_acc\_spheric ( void )

calculate and return the accel. in spherical coord

##### Returns

acceleration in spherical coord

##### See also

[TypeAccSpheric](#)

#### 7.2.3.7 int e\_read\_acc\_x ( void )

calculate and return acceleration on the x axis

##### Returns

acceleration on the x axis

#### 7.2.3.8 TypeAccRaw e\_read\_acc\_xyz ( void )

calculate and return acceleration on the x,y,z axis

##### Returns

acceleration on the x,y,z axis

##### See also

[TypeAccRaw](#)

#### 7.2.3.9 int e\_read\_acc\_y ( void )

calculate and return acceleration on the y axis

##### Returns

acceleration on the y axis



#### 7.2.3.10 int e\_read\_acc\_z( void )

calculate and return acceleration on the z axis

##### Returns

acceleration on the z axis

#### 7.2.3.11 float e\_read\_inclination( void )

calculate and return the inclination angle

##### Returns

inclination angle of the robot

#### 7.2.3.12 float e\_read\_orientation( void )

calculate and return the orientation angle

##### Returns

orientation of the accel vector

## 7.3 a\_d/advance\_ad\_scan/e\_ad\_conv.c File Reference

Module for the advance Analogic/Digital conversion.

```
#include "../motor_led/e_epuck_ports.h"
#include "e_ad_conv.h"
#include "../utility/utility.h"
#include "../motor_led/e_init_port.h"
```

### Functions

- void [e\\_init\\_ad\\_scan](#) (unsigned char only\_micro)  
*Initialize all the A/D register needed.*
- void [\\_\\_attribute\\_\\_](#) ((\_\_interrupt\_\_, auto\_psv))  
*Save the AD buffer registers in differents arrays.*
- unsigned char [e\\_ad\\_is\\_acquisition\\_completed](#) (void)  
*To know if the ADC acquisitionn is completed.*
- unsigned char [e\\_ad\\_is\\_array\\_filled](#) (void)  
*To know if the ADC acquisitionn of microphone only is completed.*
- void [e\\_ad\\_scan\\_on](#) (void)  
*Enable the ADC conversion.*
- void [e\\_ad\\_scan\\_off](#) (void)  
*Disable the ADC conversion.*

## Variables

- int `e_mic_scan` [3][MIC\_SAMP\_NB]
- int `e_acc_scan` [3][ACC\_SAMP\_NB]
- unsigned int `e_last_mic_scan_id` = 0
- unsigned int `e_last_acc_scan_id` = 0
- int `e_ambient_ir` [10]
- int `e_ambient_and_reflected_ir` [10]
- static unsigned char `is_ad_acquisition_completed` = 0
- static unsigned char `is_ad_array_filled` = 0
- static unsigned char `micro_only` = 0
- int `selector`
- unsigned char `updateAccl2CCounter` = 0
- int `centre_z`
- unsigned int `tickAdclsr` = 0

### 7.3.1 Detailed Description

Module for the advance Analogic/Digital conversion.

#### Author

Code: Francesco Mondada, Michael-Bonani & Borter Jean-Joel  
Doc: Jonathan Besuchet

### 7.3.2 Function Documentation

#### 7.3.2.1 void \_\_attribute\_\_ ( (\_\_interrupt\_\_, auto\_psv) )

Save the AD buffer registers in differents arrays.

Interrupt from timer3.

Interrupt from timer2.

Parse the chained list of agenda.

Increment counter only.

Check if agenda has to be treated according to the cycle value and current counter value.

Do it for number of cycle positive or null.

Check if a service has to be activated.

#### 7.3.2.2 unsigned char e\_ad\_is\_acquisition\_completed ( void )

To know if the ADC acquisitionn is completed.

#### Returns

0 if the new acquisition is not made, 1 if completed.

## 7.3.2.3 unsigned char e\_ad\_is\_array\_filled ( void )

To know if the ADC acquisitionn of microphone only is completed.

## Returns

0 if the new acquisition is not made, 1 if completed.

## 7.3.2.4 void e\_ad\_scan\_off ( void )

Disable the ADC conversion.

## 7.3.2.5 void e\_ad\_scan\_on ( void )

Enable the ADC conversion.

7.3.2.6 void e\_init\_ad\_scan ( unsigned char *only\_micro* )

Initialize all the A/D register needed.

Set up the different ADC register to process the AD conversion by scanning the used AD channels. Each value of the channels will be stored in a different AD buffer register and an inturrupt will occure at the end of the scan.

## Parameters

<i>only_micro</i>	Put MICRO_ONLY to use only the three microphones at 33kHz. Put ALL_ADC to use all the stuff using the ADC.
-------------------	--

## 7.3.3 Variable Documentation

## 7.3.3.1 int centre\_z

## 7.3.3.2 int e\_acc\_scan[3][ACC\_SAMP\_NB]

Array to store the acc values

## 7.3.3.3 int e\_ambient\_and\_reflected\_ir[10]

Array to store the light when IR led is on

## 7.3.3.4 int e\_ambient\_ir[10]

Array to store the ambient light measurement

7.3.3.5 unsigned int e\_last\_acc\_scan\_id = 0

7.3.3.6 unsigned int e\_last\_mic\_scan\_id = 0

7.3.3.7 int e\_mic\_scan[3][MIC\_SAMP\_NB]

Array to store the mic values

7.3.3.8 unsigned char is\_ad\_acquisition\_completed = 0 [static]

7.3.3.9 unsigned char is\_ad\_array\_filled = 0 [static]

7.3.3.10 unsigned char micro\_only = 0 [static]

7.3.3.11 int selector

7.3.3.12 unsigned int tickAdclsr = 0

7.3.3.13 unsigned char updateAccl2CCounter = 0

## 7.4 a\_d/e\_ad\_conv.c File Reference

Module for the Analogic/Digital conversion.

```
#include "../motor_led/e_epuck_ports.h"
```

### Functions

- void [e\\_init\\_ad](#) (void)  
*Initialize all the A/D register needed.*
- int [e\\_read\\_ad](#) (unsigned int channel)  
*Function to sample an AD channel.*

#### 7.4.1 Detailed Description

Module for the Analogic/Digital conversion.

##### Author

Code: Lucas Meier & Francesco Mondada, Michael Bonami  
Doc: Jonathan Besuchet

#### 7.4.2 Function Documentation

7.4.2.1 void [e\\_init\\_ad](#) ( void )

Initialize all the A/D register needed.

7.4.2.2 int [e\\_read\\_ad](#) ( unsigned int *channel* )

Function to sample an AD channel.

## Parameters

<i>channel</i>	The A/D channel you want to sample Must be between 0 to 15
----------------	--

## Returns

The sampled value on the specified channel

## 7.5 a\_d/advance\_ad\_scan/e\_ad\_conv.h File Reference

Module for the advance Analogic/Digital conversion.

## Macros

- #define `MIC_SAMP_FREQ` 16384.0
- #define `ACC_PROX_SAMP_FREQ` 256.0
- #define `PULSE_LENGTH` 0.0003
- #define `ACC_PROX_PERIOD` (int)(`MIC_SAMP_FREQ/ACC_PROX_SAMP_FREQ`)
- #define `PULSE_PERIOD` (int)(`PULSE_LENGTH*MIC_SAMP_FREQ`)
- #define `ADCS_3_CHAN` (int)( $2.0*FCY/(MIC\_SAMP\_FREQ*(14+1)*3)-1$ )
- #define `ADCS_5_CHAN` (int)( $2.0*FCY/(MIC\_SAMP\_FREQ*(14+1)*5)-1$ )
- #define `ADCS_6_CHAN` (int)( $2.0*FCY/(MIC\_SAMP\_FREQ*(14+1)*6)-1$ )
- #define `ADCS_2_CHAN` (int)( $2.0*FCY/(MIC\_SAMP\_FREQ*(14+1)*2)-1$ )
- #define `MIC_SAMP_NB` 100
- #define `ACC_SAMP_NB` 50
- #define `MICRO_ONLY` 1
- #define `ALL_ADC` 0
- #define `ADC_ISR_PERIOD_MS` 0.061

## Functions

- void `e_init_ad_scan` (unsigned char only\_micro)  
*Initialize all the A/D register needed.*
- unsigned char `e_ad_is_array_filled` (void)  
*To know if the ADC acquisitionn of microphone only is completed.*
- unsigned char `e_ad_is_acquisition_completed` (void)  
*To know if the ADC acquisitionn is completed.*
- void `e_ad_scan_on` (void)  
*Enable the ADC conversion.*
- void `e_ad_scan_off` (void)  
*Disable the ADC conversion.*

### 7.5.1 Detailed Description

Module for the advance Analogic/Digital conversion.

The advance converter module is set to operate by itself. It uses the ADC interrupt to launch the acquisitions. Then no timer is needed.

The data sampled are stored in the corresponding array:

- `e_mic_scan[3][MIC_SAMP_NB]` Array to store the mic values
- `e_acc_scan[3][ACC_SAMP_NB]` Array to store the acc values
- `e_ambient_ir[8]` Array to store ambient light measurement
- `e_ambient_and_reflected_ir[8]` Array to store ambient and reflected light measurement

In all the files of this module (`e_acc.c`, `e_micro.c`, `e_prox.c`), these arrays are declared has "extern". In this way we can access the arrays for exemple like this (function `e_get_acc` from `e_acc.c`):

```
extern int e_acc_scan[3][ACC_SAMP_NB];

int e_get_acc(unsigned int captor)
{
    if (captor < 3)
        return (e_acc_scan[captor][e_last_acc_scan_id]);
    else
        return ((int)ANGLE_ERROR);
}
```

#### Author

Code: Francesco Mondada, Michael-Bonani & Borter Jean-Joel  
 Doc: Jonathan Besuchet

### 7.5.2 Macro Definition Documentation

7.5.2.1 `#define ACC_PROX_PERIOD (int)(MIC_SAMP_FREQ/ACC_PROX_SAMP_FREQ)`

7.5.2.2 `#define ACC_PROX_SAMP_FREQ 256.0`

7.5.2.3 `#define ACC_SAMP_NB 50`

7.5.2.4 `#define ADC_ISR_PERIOD_MS 0.061`

7.5.2.5 `#define ADCS_2_CHAN (int)(2.0*FCY/(MIC_SAMP_FREQ*(14+1)*2)-1)`

7.5.2.6 `#define ADCS_3_CHAN (int)(2.0*FCY/(MIC_SAMP_FREQ*(14+1)*3)-1)`

7.5.2.7 `#define ADCS_5_CHAN (int)(2.0*FCY/(MIC_SAMP_FREQ*(14+1)*5)-1)`

7.5.2.8 `#define ADCS_6_CHAN (int)(2.0*FCY/(MIC_SAMP_FREQ*(14+1)*6)-1)`

7.5.2.9 `#define ALL_ADC 0`

7.5.2.10 `#define MIC_SAMP_FREQ 16384.0`

7.5.2.11 `#define MIC_SAMP_NB 100`

7.5.2.12 `#define MICRO_ONLY 1`

7.5.2.13 `#define PULSE_LENGTH 0.0003`

7.5.2.14 `#define PULSE_PERIOD (int)(PULSE_LENGTH*MIC_SAMP_FREQ)`

### 7.5.3 Function Documentation

7.5.3.1 `unsigned char e_ad_is_acquisition_completed ( void )`

To know if the ADC acquisitionn is completed.

#### Returns

0 if the new acquisition is not made, 1 if completed.

7.5.3.2 `unsigned char e_ad_is_array_filled ( void )`

To know if the ADC acquisitionn of microphone only is completed.

#### Returns

0 if the new acquisition is not made, 1 if completed.

7.5.3.3 `void e_ad_scan_off ( void )`

Disable the ADC conversion.

7.5.3.4 `void e_ad_scan_on ( void )`

Enable the ADC conversion.

7.5.3.5 `void e_init_ad_scan ( unsigned char only_micro )`

Initialize all the A/D register needed.

Set up the different ADC register to process the AD conversion by scanning the used AD channels. Each value of the channels will be stored in a different AD buffer register and an interrupt will occur at the end of the scan.

**Parameters**

<i>only_micro</i>	Put MICRO_ONLY to use only the three microphones at 33kHz. Put ALL_ADC to use all the stuff using the ADC.
-------------------	--

## 7.6 a\_d/e\_ad\_conv.h File Reference

Module for the Analogic/Digital conversion.

**Functions**

- void [e\\_init\\_ad](#) (void)  
*Initialize all the A/D register needed.*
- int [e\\_read\\_ad](#) (unsigned int channel)  
*Function to sample an AD channel.*

### 7.6.1 Detailed Description

Module for the Analogic/Digital conversion.

**Author**

Code: Lucas Meier & Francesco Mondada, Michael Bonami  
Doc: Jonathan Besuchet

### 7.6.2 Function Documentation

#### 7.6.2.1 void e\_init\_ad ( void )

Initialize all the A/D register needed.

#### 7.6.2.2 int e\_read\_ad ( unsigned int channel )

Function to sample an AD channel.

**Parameters**

<i>channel</i>	The A/D channel you want to sample Must be between 0 to 15
----------------	--

**Returns**

The sampled value on the specified channel



## 7.7 a\_d/advance\_ad\_scan/e\_micro.c File Reference

Accessing the microphone data.

```
#include "p30F6014A.h"  
#include "e_ad_conv.h"  
#include "../motor_led/e_epuck_ports.h"
```

### Functions

- int [e\\_get\\_micro](#) (unsigned int micro\_id)  
*Get the last value of a given micro.*
- int [e\\_get\\_micro\\_average](#) (unsigned int micro\_id, unsigned int filter\_size)  
*Get the average on a given number of sample from a micro.*
- int [e\\_get\\_micro\\_volume](#) (unsigned int micro\_id)  
*Get the difference between the highest and lowest sample. Beware that from HWRev 1.3 the microphone sensitivity resulted a little bit different from the previous hardware revision; some empirical tests show that the difference is about +/-15%.*
- void [e\\_get\\_micro\\_last\\_values](#) (int micro\_id, int \*result, unsigned samples\_nb)  
*Write to a given array, the last values of one micro.*

### Variables

- int [e\\_mic\\_scan](#) [3][MIC\_SAMP\_NB]
- unsigned int [e\\_last\\_mic\\_scan\\_id](#)

#### 7.7.1 Detailed Description

Accessing the microphone data.

The functions of this file are made to deal with the microphones data. You can simply get the current value of a given microphone. You can know the volume of noise that the e-puck is enduring. You can average the signal with a specified size.

#### Author

Code: Borter Jean-Joel  
Doc: Jonathan Besuchet

#### 7.7.2 Function Documentation

##### 7.7.2.1 int e\_get\_micro ( unsigned int *micro\_id* )

Get the last value of a given micro.

## Parameters

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <a href="#">MIC0_BUFFER</a> , <a href="#">MIC1_BUFFER</a> , <a href="#">MIC2_BUFFER</a> defined in <a href="#">e_micro.h</a> )
-----------------	---

## Returns

result last value of the micro

7.7.2.2 int e\_get\_micro\_average ( unsigned int *micro\_id*, unsigned int *filter\_size* )

Get the average on a given number of sample from a micro.

## Parameters

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <a href="#">MIC0_BUFFER</a> , <a href="#">MIC1_BUFFER</a> , <a href="#">MIC2_BUFFER</a> defined in <a href="#">e_micro.h</a> )
<i>filter_size</i>	number of sample to average

## Returns

result last value of the micro

7.7.2.3 void e\_get\_micro\_last\_values ( int *micro\_id*, int \* *result*, unsigned *samples\_nb* )

Write to a given array, the last values of one micro.

Write to a given array, the last values of one micro. The values are stored with the last one first, and the older one at the end of the array.

[ t ][ t-1 ][ t-2 ][ t-3 ]...[ t-(samples\_nb-1) ][ t-samples\_nb ]

## Parameters

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <a href="#">MIC0_BUFFER</a> , <a href="#">MIC1_BUFFER</a> , <a href="#">MIC2_BUFFER</a> defined in <a href="#">e_micro.h</a> )
* <i>result</i>	pointer on the result array
<i>samples_nb</i>	size of the result array (must be between 1 and <a href="#">MIC_SAMP_NB</a> )

7.7.2.4 int e\_get\_micro\_volume ( unsigned int *micro\_id* )

Get the difference between the highest and lowest sample. Beware that from HWRev 1.3 the microphone sensitivity resulted a little bit different from the previous hardware revision; some empirical tests show that the difference is about +/-15%.

## Parameters

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <a href="#">MIC0_BUFFER</a> , <a href="#">MIC1_BUFFER</a> , <a href="#">MIC2_BUFFER</a> defined in <a href="#">e_micro.h</a> )
-----------------	---

**Returns**

result volume

**7.7.3 Variable Documentation****7.7.3.1 unsigned int e\_last\_mic\_scan\_id****7.7.3.2 int e\_mic\_scan[3][MIC\_SAMP\_NB]**

Array to store the mic values

**7.8 a\_d/e\_micro.c File Reference**

Accessing the microphone data.

```
#include "e_ad_conv.h"
#include "../motor_led/e_epuck_ports.h"
#include "e_micro.h"
```

**Functions**

- void [e\\_init\\_micro](#) (void)  
*Init the microphone A/D converter.*
- void [e\\_get\\_micro](#) (int \*m0, int \*m1, int \*m2)  
*To get the m0, m1, m2 microphones's values.*

**7.8.1 Detailed Description**

Accessing the microphone data.

A little exemple which takes the volume of micro1 and if the sound level is more than 2000. The LED1 is turned on.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/e_micro.h>

int main(void)
{
    int m1, m2, m3;
    e_init_port();
    e_init_micro();
    while(1)
    {
        long i;
        e_get_micro(&m1, &m2, &m3);
        if(m1 < 2000)
            LED0 = 0;
        else
            LED0 = 1;
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

**Author**

Code: Michael Bonani  
Doc: Jonathan Besuchet

## 7.8.2 Function Documentation

### 7.8.2.1 void e\_get\_micro ( int \* m0, int \* m1, int \* m2 )

To get the m0, m1, m2 microphones's values.

#### Parameters

<i>m0</i>	A pointer to store the m0 analogic value
<i>m1</i>	A pointer to store the m1 analogic value
<i>m2</i>	A pointer to store the m2 analogic value

### 7.8.2.2 void e\_init\_micro ( void )

Init the microphone A/D converter.

#### Warning

Must be called before starting using microphone

## 7.9 a\_d/advance\_ad\_scan/e\_micro.h File Reference

Accessing the microphone data.

### Macros

- `#define MIC0_BUFFER 0`
- `#define MIC1_BUFFER 1`
- `#define MIC2_BUFFER 2`

### Functions

- int `e_get_micro` (unsigned int micro\_id)  
*Get the last value of a given micro.*
- int `e_get_micro_average` (unsigned int micro\_id, unsigned int filter\_size)  
*Get the average on a given number of sample from a micro.*
- int `e_get_micro_volume` (unsigned int micro\_id)  
*Get the difference between the highest and lowest sample. Beware that from HWRev 1.3 the microphone sensitivity resulted a little bit different from the previous hardware revision; some empirical tests show that the difference is about +/-15%.*

## 7.9.1 Detailed Description

Accessing the microphone data.

The functions of this file are made to deal with the microphones data. You can simply get the current value of a given microphone. You can know the volume of noise that the e-puck is enduring. You can average the signal with a specified size.

### Author

Code: Borter Jean-Joel  
Doc: Jonathan Besuchet

## 7.9.2 Macro Definition Documentation

7.9.2.1 `#define MIC0_BUFFER 0`

7.9.2.2 `#define MIC1_BUFFER 1`

7.9.2.3 `#define MIC2_BUFFER 2`

## 7.9.3 Function Documentation

7.9.3.1 `int e_get_micro ( unsigned int micro_id )`

Get the last value of a given micro.

### Parameters

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <a href="#">MIC0_BUFFER</a> , <a href="#">MIC1_BUFFER</a> , <a href="#">MIC2_BUFFER</a> defined in e_micro.h)
-----------------	--

### Returns

result last value of the micro

7.9.3.2 `int e_get_micro_average ( unsigned int micro_id, unsigned int filter_size )`

Get the average on a given number of sample from a micro.

### Parameters

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <a href="#">MIC0_BUFFER</a> , <a href="#">MIC1_BUFFER</a> , <a href="#">MIC2_BUFFER</a> defined in e_micro.h)
<i>filter_size</i>	number of sample to average

**Returns**

result last value of the micro

**7.9.3.3 int e\_get\_micro\_volume ( unsigned int *micro\_id* )**

Get the difference between the highest and lowest sample. Beware that from HWRev 1.3 the microphone sensitivity resulted a little bit different from the previous hardware revision; some empirical tests show that the difference is about +/-15%.

**Parameters**

<i>micro_id</i>	micro's ID (0, 1, or 2) (use <code>MIC0_BUFFER</code> , <code>MIC1_BUFFER</code> , <code>MIC2_BUFFER</code> defined in <code>e_micro.h</code> )
-----------------	---

**Returns**

result volume

**7.10 a\_d/e\_micro.h File Reference**

Accessing the microphone data.

**Functions**

- void `e_init_micro` (void)  
*Init the microphone A/D converter.*
- void `e_get_micro` (int \*m1, int \*m2, int \*m3)  
*To get the m0, m1, m2 microphones's values.*

**7.10.1 Detailed Description**

Accessing the microphone data.

A little exemple which takes the volume of micro1 and if the sound level is more than 2000. The LED1 is turned on.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/e_micro.h>

int main(void)
{
    int m1, m2, m3;
    e_init_port ();
    e_init_micro ();
    while(1)
    {
        long i;
        e_get_micro(&m1, &m2, &m3);
        if(m1 < 2000)
            LED0 = 0;
        else
            LED0 = 1;
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

**Author**

Code: Michael Bonani  
Doc: Jonathan Besuchet

## 7.10.2 Function Documentation

### 7.10.2.1 void e\_get\_micro ( int \* m0, int \* m1, int \* m2 )

To get the m0, m1, m2 microphones's values.

#### Parameters

<i>m0</i>	A pointer to store the m0 analogic value
<i>m1</i>	A pointer to store the m1 analogic value
<i>m2</i>	A pointer to store the m2 analogic value

### 7.10.2.2 void e\_init\_micro ( void )

Init the microphone A/D converter.

#### Warning

Must be called before starting using microphone

## 7.11 a\_d/advance\_ad\_scan/e\_prox.c File Reference

Accessing proximity sensor of e-puck.

```
#include "e_ad_conv.h"
#include "../motor_led/e_epuck_ports.h"
#include "e_prox.h"
```

### Functions

- void [e\\_calibrate\\_ir](#) ()  
*To calibrate your ir sensor.*
- int [e\\_get\\_prox](#) (unsigned int sensor\_number)  
*To get the analogic proxy sensor value of a specific ir sensor.*
- int [e\\_get\\_calibrated\\_prox](#) (unsigned int sensor\_number)  
*To get the calibrated value of the ir sensor.*
- int [e\\_get\\_ambient\\_light](#) (unsigned int sensor\_number)  
*To get the analogic ambient light value of a specific ir sensor.*

### Variables

- int [e\\_ambient\\_ir](#) [10]
- int [e\\_ambient\\_and\\_reflected\\_ir](#) [10]
- static int [init\\_value\\_ir](#) [10] = {0,0,0,0,0,0,0,0,0,0}

### 7.11.1 Detailed Description

Accessing proximity sensor of e-puck.

The functions of this file are made to deal with the proximity data. You can know the value of the ambient light detected by the sensor. You can estimate the distance between the e-puck and an obstacle by using `e_get_prox(unsigned int sensor_number)` function.

A little exemple which turn the LED0 when an obstacle is detected by the proximity sensor number 0.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/advance_ad_scan/e_prox.h>
#include <a_d/advance_ad_scan/e_ad_conv.h>

int main(void)
{
    int proxy0;
    e_init_port ();
    e_init_ad_scan();
    while(1)
    {
        long i;
        proxy0 = e_get_prox(0);
        if(proxy0 < 1000)
            LED0 = 0;
        else
            LED0 = 1;
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

#### Author

Code: Lucas Meier

Doc: Jonathan Besuchet

### 7.11.2 Function Documentation

#### 7.11.2.1 void e\_calibrate\_ir ( )

To calibrate your ir sensor.

#### Warning

Call this function one time before calling `e_get_calibrated_prox`

#### 7.11.2.2 int e\_get\_ambient\_light ( unsigned int *sensor\_number* )

To get the analogic ambient light value of a specific ir sensor.

This function return the analogic value of the ambient light measurement.

#### Parameters

<code>sensor_number</code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------------	--



**Returns**

The analogic value of the specified proxy sensor

**7.11.2.3 int e\_get\_calibrated\_prox ( unsigned int *sensor\_number* )**

To get the calibrated value of the ir sensor.

This function return the calbrated analogic value of the ir sensor.

**Warning**

Befroe using this function you have to calibrate your ir sensor (only one time) by calling [e\\_calibrate\\_ir\(\)](#).

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the calibrated value. Must be between 0 to 7.
----------------------	---

**Returns**

The analogic value of the specified proxy sensor

**7.11.2.4 int e\_get\_prox ( unsigned int *sensor\_number* )**

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proxymity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate:  $\text{reflected light} = (\text{reflected light} + \text{ambient light}) - \text{ambient light}$
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obsacle is near.

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

**Returns**

The analogic value of the specified proxy sensor

### 7.11.3 Variable Documentation

#### 7.11.3.1 `int e_ambient_and_reflected_ir[10]`

Array to store the light when IR led is on

#### 7.11.3.2 `int e_ambient_ir[10]`

Array to store the ambient light measurement

#### 7.11.3.3 `int init_value_ir[10] = {0,0,0,0,0,0,0,0,0,0} [static]`

## 7.12 `a_d/e_prox.c` File Reference

Accessing proximity sensor of e-puck with timer 1.

```
#include "e_ad_conv.h"
#include "../motor_led/e_epuck_ports.h"
#include "e_prox.h"
```

### Functions

- void `init_tmr1` (void)
- void `__attribute__` ((interrupt, auto\_psv))
  - The VSYNC interrupt. This interrupt is called every time the Vertical sync signal is asserted This mean that the picture is coming from the camera ( we will have the first line soon )*
- void `e_init_prox` (void)
  - Init the proxymity sensor A/D converter and the timer1.*
- void `e_stop_prox` (void)
  - Stop the acquisition (stop timer1)*
- int `e_get_prox` (unsigned int sensor\_number)
  - To get the analogic proxy sensor value of a specific sensor.*
- int `e_get_ambient_light` (unsigned int sensor\_number)
  - To get the analogic ambient light value of a specific sensor.*

### Variables

- static int `ambient_ir` [8]
- static int `ambient_and_reflected_ir` [8]
- static int `reflected_ir` [8]

## 7.12.1 Detailed Description

Accessing proximity sensor of e-puck with timer 1.

The functions of this file are made to deal with the proximity sensor data. You can know the value of the ambient light detected by the sensor. You can estimate the distance between the e-puck and an obstacle by using `e_get_prox(unsigned int sensor_number)` function.

A little exemple which turn the LED0 when an obstacle is detected by the proximity sensor number 0.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/e_prox.h>

int main(void)
{
    int value;
    e_init_port();
    e_init_prox();
    while(1)
    {
        long i;
        value = e_get_prox(0);
        if(value > 1000) //LED0 on if an obstacle is detected by proxy0
            LED0 = 1;
        else
            LED0 = 0;
        for(i=0; i<1000000; i++) { asm("nop"); }
    }
}
```

### Warning

This module uses the timer1

### Author

Code: Lucas Meier & Francesco Mondada, Michael Bonani  
Doc: Jonathan Besuchet

## 7.12.2 Function Documentation

### 7.12.2.1 void \_\_attribute\_\_((interrupt, auto\_psv))

The VSYNC interrupt. This interrupt is called every time the Vertical sync signal is asserted This mean that the picture is coming from the camera ( we will have the first line soon )

The Pixel Clock interrupt. This interrupt is called every time the Pixel clock signal is asserted This mean that the next byte is ready to be read.

### 7.12.2.2 int e\_get\_ambient\_light ( unsigned int sensor\_number )

To get the analogic ambient light value of a specific sensor.

To get the analogic ambient light value of a specific ir sensor.

This function retur the analogic value of the ambient light measurement.

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

**Returns**

The analogic value of the specified proxy sensor

**7.12.2.3 int e\_get\_prox ( unsigned int *sensor\_number* )**

To get the analogic proxy sensor value of a specific sensor.

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proximity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate: reflected light = (reflected light + ambient light) - ambient light
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obsacle is near.

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

**Returns**

The analogic value of the specified proxy sensor

**7.12.2.4 void e\_init\_prox ( void )**

Init the proximity sensor A/D converter and the timer1.

**Warning**

Must be called before starting using proximity sensor

**7.12.2.5 void e\_stop\_prox ( void )**

Stop the acquisition (stop timer1)

7.12.2.6 void init\_tmr1 ( void )

## 7.12.3 Variable Documentation

7.12.3.1 int ambient\_and\_reflected\_ir[8] [static]

7.12.3.2 int ambient\_ir[8] [static]

7.12.3.3 int reflected\_ir[8] [static]

## 7.13 a\_d/advance\_ad\_scan/e\_prox.h File Reference

Accessing proximity sensor of e-puck.

### Functions

- void [e\\_calibrate\\_ir](#) ()  
*To calibrate your ir sensor.*
- int [e\\_get\\_prox](#) (unsigned int sensor\_number)  
*To get the analogic proxy sensor value of a specific ir sensor.*
- int [e\\_get\\_calibrated\\_prox](#) (unsigned int sensor\_number)  
*To get the calibrated value of the ir sensor.*
- int [e\\_get\\_ambient\\_light](#) (unsigned int sensor\_number)  
*To get the analogic ambient light value of a specific ir sensor.*

### 7.13.1 Detailed Description

Accessing proximity sensor of e-puck.

The functions of this file are made to deal with the proximity data. You can know the value of the ambient light detected by the sensor. You can estimate the distance between the e-puck and an obstacle by using [e\\_get\\_prox](#) function.

A little exemple which turn the LED0 when an obstacle is detected by the proximity sensor number 0.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/advance_ad_scan/e_prox.h>
#include <a_d/advance_ad_scan/e_ad_conv.h>

int main(void)
{
    int proxy0;
    e_init_port ();
    e_init_ad_scan();
    while(1)
    {
        long i;
        proxy0 = e_get_prox(0);
        if(proxy0 < 1000)
            LED0 = 0;
        else
            LED0 = 1;
        for(i=0; i<1000000; i++) { asm("nop"); }
    }
}
```

### Author

Code: Lucas Meier

Doc: Jonathan Besuchet

## 7.13.2 Function Documentation

### 7.13.2.1 void e\_calibrate\_ir ( )

To calibrate your ir sensor.

#### Warning

Call this function one time before calling e\_get\_calibrated\_prox

### 7.13.2.2 int e\_get\_ambient\_light ( unsigned int *sensor\_number* )

To get the analogic ambient light value of a specific ir sensor.

This function return the analogic value of the ambient light measurement.

#### Parameters

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

#### Returns

The analogic value of the specified proxy sensor

To get the analogic ambient light value of a specific ir sensor.

This function retur the analogic value of the ambient light measurement.

#### Parameters

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

#### Returns

The analogic value of the specified proxy sensor

### 7.13.2.3 int e\_get\_calibrated\_prox ( unsigned int *sensor\_number* )

To get the calibrated value of the ir sensor.

This function return the calibrated analogic value of the ir sensor.

#### Warning

Befroe using this function you have to calibrate your ir sensor (only one time) by calling [e\\_calibrate\\_ir\(\)](#).

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the calibrated value. Must be between 0 to 7.
----------------------	---

**Returns**

The analogic value of the specified proxy sensor

**7.13.2.4 int e\_get\_prox ( unsigned int *sensor\_number* )**

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proximity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate:  $\text{reflected light} = (\text{reflected light} + \text{ambient light}) - \text{ambient light}$
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obsacle is near.

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

**Returns**

The analogic value of the specified proxy sensor

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proximity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate:  $\text{reflected light} = (\text{reflected light} + \text{ambient light}) - \text{ambient light}$
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obsacle is near.

**Parameters**

<code>sensor_number</code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------------	--

**Returns**

The analogic value of the specified proxy sensor

## 7.14 a\_d/e\_prox.h File Reference

Accessing proximity sensor of e-puck with timer 1.

**Functions**

- void `e_init_prox` (void)  
*Init the proxymity sensor A/D converter and the timer1.*
- void `e_stop_prox` (void)  
*Stop the acquisition (stop timer1)*
- int `e_get_prox` (unsigned int sensor\_number)  
*To get the analogic proxy sensor value of a specific ir sensor.*
- int `e_get_ambient_light` (unsigned int sensor\_number)  
*To get the analogic ambient light value of a specific ir sensor.*

### 7.14.1 Detailed Description

Accessing proximity sensor of e-puck with timer 1.

The functions of this file are made to deal with the proximity sensor data. You can know the value of the ambient light detected by the sensor. You can estimate the distance between the e-puck and an obstacle by using `e_get_prox(unsigned int sensor_number)` function.

A little exemple which turn the LED0 when an obstacle is detected by the proximity sensor number 0.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/e_prox.h>

int main(void)
{
    int value;
    e_init_port ();
    e_init_prox ();
    while(1)
    {
        long i;
        value = e_get_prox(0);
        if(value > 1000) //LED0 on if an obstacle is detected by proxy0
            LED0 = 1;
        else
            LED0 = 0;
        for(i=0; i<1000000; i++) { asm("nop"); }
    }
}
```

**Warning**

This module uses the timer1

**Author**

Code: Lucas Meier & Francesco Mondada, Michael Bonani  
Doc: Jonathan Besuchet



## 7.14.2 Function Documentation

### 7.14.2.1 `int e_get_ambient_light ( unsigned int sensor_number )`

To get the analogic ambient light value of a specific ir sensor.

This function return the analogic value of the ambient light measurement.

#### Parameters

<code><i>sensor_number</i></code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
-----------------------------------	--

#### Returns

The analogic value of the specified proxy sensor

To get the analogic ambient light value of a specific ir sensor.

This function retur the analogic value of the ambient light measurement.

#### Parameters

<code><i>sensor_number</i></code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
-----------------------------------	--

#### Returns

The analogic value of the specified proxy sensor

### 7.14.2.2 `int e_get_prox ( unsigned int sensor_number )`

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proximity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate:  $\text{reflected light} = (\text{reflected light} + \text{ambient light}) - \text{ambient light}$
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obsacle is near.

#### Parameters

<code><i>sensor_number</i></code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
-----------------------------------	--

**Returns**

The analogic value of the specified proxy sensor

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proximity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate: reflected light = (reflected light + ambient light) - ambient light
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obsacle is near.

**Parameters**

<i>sensor_number</i>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------	--

**Returns**

The analogic value of the specified proxy sensor

**7.14.2.3 void e\_init\_prox ( void )**

Init the proximity sensor A/D converter and the timer1.

**Warning**

Must be called before starting using proximity sensor

Init the proximity sensor A/D converter and the timer1.

**Warning**

Must be called before starting using proximity sensor

**7.14.2.4 void e\_stop\_prox ( void )**

Stop the acquisition (stop timer1)

Stop the acquisition (stop timer1)

## 7.15 a\_d/e\_accelerometer.c File Reference

Accessing the accelerometer sensor data.

```
#include "e_ad_conv.h"
#include "../motor_led/e_epuck_ports.h"
#include "e_accelerometer.h"
```

### Functions

- void `e_init_acc` (void)  
*Init the accelerometer A/D converter.*
- void `e_get_acc` (int \*x, int \*y, int \*z)  
*To get the analogic x, y, z axis accelerations.*

### 7.15.1 Detailed Description

Accessing the accelerometer sensor data.

A little exemple to read the accelerator.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/e_accelerometer.h>

int main(void)
{
    int x, y, z;
    e_init_port();
    e_init_acc();
    while(1)
    {
        long i;
        e_get_acc(&x, &y, &z);
        if(z < 2100) //LED4 on if e-puck is on the back
        {
            LED0 = 0;
            LED4 = 1;
        }
        else //LED0 on if e-puck is on his wells
        {
            LED0 = 1;
            LED4 = 0;
        }
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

### Author

Code: Michael Bonani  
Doc: Jonathan Besuchet

### 7.15.2 Function Documentation

#### 7.15.2.1 void e\_get\_acc ( int \* x, int \* y, int \* z )

To get the analogic x, y, z axis accelerations.

**Parameters**

x	A pointer to store the analogic x acceleration
y	A pointer to store the analogic y acceleration
z	A pointer to store the analogic z acceleration

**7.15.2.2 void e\_init\_acc ( void )**

Init the accelerometer A/D converter.

**Warning**

Must be called before starting using accelerometer

**7.16 a\_d/e\_accelerometer.h File Reference**

Accessing the accelerometer sensor data.

**Functions**

- void `e_init_acc` (void)  
*Init the accelerometer A/D converter.*
- void `e_get_acc` (int \*x, int \*y, int \*z)  
*To get the analogic x, y, z axis accelerations.*

**7.16.1 Detailed Description**

Accessing the accelerometer sensor data.

A little exemple to read the accelerator.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/e_accelerometer.h>

int main(void)
{
    int x, y, z;
    e_init_port();
    e_init_acc();
    while(1)
    {
        long i;
        e_get_acc(&x, &y, &z);
        if(z < 2100) //LED4 on if e-puck is on the back
        {
            LED0 = 0;
            LED4 = 1;
        }
        else //LED0 on if e-puck is on his wells
        {
            LED0 = 1;
            LED4 = 0;
        }
        for(i=0; i<100000; i++) { asm("nop"); }
    }
}
```

**Author**

Code: Michael Bonani  
Doc: Jonathan Besuchet

## 7.16.2 Function Documentation

### 7.16.2.1 void e\_get\_acc ( int \* x, int \* y, int \* z )

To get the analogic x, y, z axis accelerations.

#### Parameters

x	A pointer to store the analogic x acceleration
y	A pointer to store the analogic y acceleration
z	A pointer to store the analogic z acceleration

### 7.16.2.2 void e\_init\_acc ( void )

Init the accelerometer A/D converter.

#### Warning

Must be called before starting using accelerometer

## 7.17 a\_d/e\_prox\_timer2.c File Reference

Control proximity sensor of e-puck with timer2.

```
#include "e_ad_conv.h"
#include "../motor_led/e_epuck_ports.h"
#include "e_prox.h"
```

### Functions

- void [init\\_tmr2](#) (void)
- void [\\_\\_attribute\\_\\_](#) ((interrupt, auto\_psv, shadow))
- void [e\\_init\\_prox](#) (void)
  - Init the proxymity sensor A/D converter and the timer2.*
- void [e\\_stop\\_prox](#) (void)
  - Stop the acquisition (stop timer2)*
- int [e\\_get\\_prox](#) (unsigned int sensor\_number)
  - To get the analogic proxy sensor value of a specific sensor.*
- int [e\\_get\\_ambient\\_light](#) (unsigned int sensor\_number)
  - To get the analogic ambient light value of a specific sensor.*

### Variables

- static int [ambient\\_ir](#) [8]
- static int [ambient\\_and\\_reflected\\_ir](#) [8]
- static int [reflected\\_ir](#) [8]

### 7.17.1 Detailed Description

Control proximity sensor of e-puck with timer2.

The functions of this file are made to deal with the proximity data. You can know the value of the ambient light detected by the sensor. You can estimate the distance between the e-puck and an obstacle by using `e_get_↔prox(unsigned int sensor_number)` function.

#### Warning

The module uses the timer2

#### See also

`e_prox.c`, `e_prox.h` to use the timer1

#### Author

Code: Lucas Meier & Francesco Mondada, Michael Bonani, Xavier Raemy  
Doc: Jonathan Besuchet

### 7.17.2 Function Documentation

7.17.2.1 `void __attribute__ ( (interrupt, auto_psv, shadow) )`

7.17.2.2 `int e_get_ambient_light ( unsigned int sensor_number )`

To get the analogic ambient light value of a specific sensor.

To get the analogic ambient light value of a specific ir sensor.

This function retur the analogic value of the ambient light measurement.

#### Parameters

<code><i>sensor_number</i></code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
-----------------------------------	--

#### Returns

The analogic value of the specified proxy sensor

7.17.2.3 `int e_get_prox ( unsigned int sensor_number )`

To get the analogic proxy sensor value of a specific sensor.

To get the analogic proxy sensor value of a specific ir sensor.

To estimate the proxymity of an obstacle, we do the following things:

- measure the ambient light
- turn on the IR led of the sensor
- measure the reflected light + ambient light
- calculate: reflected light = (reflected light + ambient light) - ambient light
- turn off the IR led of the sensor

The result value of this function is: reflected light. More this value is great, more the obstacle is near.

#### Parameters

<code>sensor_number</code>	The proxy sensor's number that you want the value. Must be between 0 to 7.
----------------------------	--

#### Returns

The analogic value of the specified proxy sensor

#### 7.17.2.4 void e\_init\_prox ( void )

Init the proximity sensor A/D converter and the timer2.

Init the proximity sensor A/D converter and the timer1.

#### Warning

Must be called before starting using proximity sensor

#### 7.17.2.5 void e\_stop\_prox ( void )

Stop the acquisition (stop timer2)

Stop the acquisition (stop timer1)

#### 7.17.2.6 void init\_tmr2 ( void )

### 7.17.3 Variable Documentation

7.17.3.1 int ambient\_and\_reflected\_ir[8] [static]

7.17.3.2 int ambient\_ir[8] [static]

7.17.3.3 int reflected\_ir[8] [static]

## 7.18 acc\_gyro/e\_lsm330.c File Reference

Manage LSM330 (accelerometer + gyro) registers.

```
#include "../I2C/e_I2C_protocol.h"
#include "e_lsm330.h"
#include <uart/e_uart_char.h>
#include "string.h"
#include "stdio.h"
```

## Macros

- #define `ACC_ADDR` 0x3C
- #define `GYRO_ADDR` 0xD4

## Functions

- void `writeReg` (unsigned char addr, unsigned char reg, unsigned char value)  
*Set the register reg to value.*
- unsigned char `readReg` (unsigned char addr, unsigned char reg)  
*Read the register reg.*
- char `readRegMulti` (char device\_add, unsigned char \*read\_buffer, char start\_address, unsigned char num← Bytes)  
*Continuous read (multiple register reading).*
- void `initAccAndGyro` (void)  
*Configure and turn on the device.*
- void `getAllAxesAccRaw` (unsigned char \*arr)
- void `getAllAxesAcc` (signed int \*x, signed int \*y, signed int \*z)
- int `getXAxisAcc` ()
- int `getYAxisAcc` ()
- int `getZAxisAcc` ()
- void `getAllAxesGyroRaw` (unsigned char \*arr)
- void `getAllAxesGyro` (signed int \*x, signed int \*y, signed int \*z)
- int `getXAxisGyro` ()
- int `getYAxisGyro` ()
- int `getZAxisGyro` ()
- signed char `getTemperature` (void)
- void `calibrateGyroscope` (int numSamples)
- float `rawToDps` (int value)  
*Raw value to degrees per second. Take as input the 2's complement value received by the device for a certain axis and return the degrees per second (knowing the gyroscope is configured with sensitivity of +- 250 dps).*
- float `rawToDpms` (int value)  
*Raw value to degrees per millisecond. Take as input the 2's complement value received by the device for a certain axis and return the degrees per millisecond (knowing the gyroscope is configured with sensitivity of +- 250 dps).*

## Variables

- signed int `gyroOffsetX` = 0
- signed int `gyroOffsetY` = 0
- signed int `gyroOffsetZ` = 0

### 7.18.1 Detailed Description

Manage LSM330 (accelerometer + gyro) registers.

#### Author

Stefano Morgani

x, y axes representation on HWRev < 1.3 (analog accelerometer MMA7260): FORWARD ^ | (-y)

- - > (+x) RIGHT

y, y axes representation on HWRev 1.3 (i2c device LSM330): FORWARD ^ | (+x)

- - > (-y) RIGHT

Thus the x and y axes are exchanged to maintain compatibility.



## 7.18.2 Macro Definition Documentation

7.18.2.1 `#define ACC_ADDR 0x3C`

7.18.2.2 `#define GYRO_ADDR 0xD4`

## 7.18.3 Function Documentation

7.18.3.1 `void calibrateGyroscope ( int numSamples )`

7.18.3.2 `void getAllAxesAcc ( signed int * x, signed int * y, signed int * z )`

7.18.3.3 `void getAllAxesAccRaw ( unsigned char * arr )`

7.18.3.4 `void getAllAxesGyro ( signed int * x, signed int * y, signed int * z )`

7.18.3.5 `void getAllAxesGyroRaw ( unsigned char * arr )`

7.18.3.6 `signed char getTemperature ( void )`

7.18.3.7 `int getXAxisAcc ( )`

7.18.3.8 `int getXAxisGyro ( )`

7.18.3.9 `int getYAxisAcc ( )`

7.18.3.10 `int getYAxisGyro ( )`

7.18.3.11 `int getZAxisAcc ( )`

7.18.3.12 `int getZAxisGyro ( )`

7.18.3.13 `void initAccAndGyro ( void )`

Configure and turn on the device.

7.18.3.14 `float rawToDpms ( int value )`

Raw value to degrees per millisecond. Take as input the 2's complement value received by the device for a certain axis and return the degrees per millisecond (knowing the gyroscope is configured with sensitivity of +- 250 dps).

7.18.3.15 `float rawToDps ( int value )`

Raw value to degrees per second. Take as input the 2's complement value received by the device for a certain axis and return the degrees per second (knowing the gyroscope is configured with sensitivity of +- 250 dps).

7.18.3.16 `unsigned char readReg ( unsigned char addr, unsigned char reg )`

Read the register reg.

**Parameters**

<i>addr</i>	The device address
<i>reg</i>	The register address

**Returns**

The register value

7.18.3.17 `char readRegMulti ( char device_add, unsigned char * read_buffer, char start_address, unsigned char numBytes )`

Continuous read (multiple register reading).

**Parameters**

<i>device_add</i>	The device address
<i>read_buffer</i>	the buffer that will contain the values read from the registers
<i>reg</i>	The register start address

**Returns**

1 to confirm the operation and 0 for an error

7.18.3.18 `void writeReg ( unsigned char addr, unsigned char reg, unsigned char value )`

Set the register *reg* to *value*.

**Parameters**

<i>addr</i>	The device address
<i>reg</i>	The register address
<i>value</i>	The value to write

**7.18.4 Variable Documentation**

7.18.4.1 `signed int gyroOffsetX = 0`

7.18.4.2 `signed int gyroOffsetY = 0`

7.18.4.3 `signed int gyroOffsetZ = 0`

**7.19 acc\_gyro/e\_lsm330.h File Reference**

LSM330 library header.

## Functions

- void `initAccAndGyro` (void)  
*Configure and turn on the device.*
- void `getAllAxesAccRaw` (unsigned char \*arr)
- void `getAllAxesAcc` (signed int \*x, signed int \*y, signed int \*z)
- void `getAllAxesGyroRaw` (unsigned char \*arr)
- void `getAllAxesGyro` (signed int \*x, signed int \*y, signed int \*z)
- signed char `getTemperature` (void)
- int `getXAxisAcc` ()
- int `getYAxisAcc` ()
- int `getZAxisAcc` ()
- int `getXAxisGyro` ()
- int `getYAxisGyro` ()
- int `getZAxisGyro` ()
- void `calibrateGyroscope` ()
- float `rawToDps` (int value)  
*Raw value to degrees per second. Take as input the 2's complement value received by the device for a certain axis and return the degrees per second (knowing the gyroscope is configured with sensitivity of +- 250 dps).*
- float `rawToDpms` (int value)  
*Raw value to degrees per millisecond. Take as input the 2's complement value received by the device for a certain axis and return the degrees per millisecond (knowing the gyroscope is configured with sensitivity of +- 250 dps).*

### 7.19.1 Detailed Description

LSM330 library header.

#### Author

Stefano Morgani

### 7.19.2 Function Documentation

7.19.2.1 void `calibrateGyroscope` ( )

7.19.2.2 void `getAllAxesAcc` ( signed int \* x, signed int \* y, signed int \* z )

7.19.2.3 void `getAllAxesAccRaw` ( unsigned char \* arr )

7.19.2.4 void `getAllAxesGyro` ( signed int \* x, signed int \* y, signed int \* z )

7.19.2.5 void `getAllAxesGyroRaw` ( unsigned char \* arr )

7.19.2.6 signed char `getTemperature` ( void )

7.19.2.7 int `getXAxisAcc` ( )

7.19.2.8 int `getXAxisGyro` ( )

7.19.2.9 int getYAxisAcc ( )

7.19.2.10 int getYAxisGyro ( )

7.19.2.11 int getZAxisAcc ( )

7.19.2.12 int getZAxisGyro ( )

7.19.2.13 void initAccAndGyro ( void )

Configure and turn on the device.

7.19.2.14 float rawToDpms ( int value )

Raw value to degrees per millisecond. Take as input the 2's complement value received by the device for a certain axis and return the degrees per millisecond (knowing the gyroscope is configured with sensitivity of +- 250 dps).

7.19.2.15 float rawToDps ( int value )

Raw value to degrees per second. Take as input the 2's complement value received by the device for a certain axis and return the degrees per second (knowing the gyroscope is configured with sensitivity of +- 250 dps).

## 7.20 bluetooth/e\_bluetooth.c File Reference

Manage Bluetooth.

```
#include "../uart/e_uart_char.h"
#include "../motor_led/e_epuck_ports.h"
#include "e_bluetooth.h"
#include "stdio.h"
#include "string.h"
#include "stdlib.h"
```

### Functions

- int [e\\_bt\\_find\\_epuck](#) (void)  
*Try to find other e-puck.*
- char [e\\_bt\\_connect\\_epuck](#) (void)
- int [e\\_bt\\_reset](#) (void)  
*Reset the bluetooth module.*
- char [e\\_bt\\_factory\\_reset](#) (void)  
*Factory reset of the bluetooth module.*
- char [e\\_bt\\_tranparent\\_mode](#) (void)  
*Change to transparent mode.*
- void [e\\_bt\\_exit\\_tranparent\\_mode](#) (void)  
*Exit from the transparent mode.*

- char `e_bt_read_local_pin_number` (char \*PIN)  
*Read the PIN number of this e-puck's bluetooth module.*
- char `e_bt_read_local_name` (char \*name)  
*Read the name of this e-puck's bluetooth module.*
- char `e_bt_write_local_pin_number` (char \*PIN)  
*Write the PIN number on this e-puck's bluetooth module.*
- char `e_bt_write_local_name` (char \*name)  
*Write the name on this e-puck's bluetooth module.*
- int `e_bt_inquiry` (struct `BtDevice` \*device)  
*Research all the accessible bluetooth devices.*
- char `e_bt_get_friendly_name` (struct `BtDevice` \*device)  
*To get the friendly name of a bluetooth device.*
- char `e_bt_establish_SPP_link` (char \*address)  
*Try to connect to another bluetooth device.*
- char `e_bt_release_SPP_link` (void)  
*Unconnect from the current bluetooth device.*
- char `e_bt_send_SPP_data` (char \*data, char datalength)  
*Send data to the current bluetooth device.*
- char `e_bt_list_local_paired_device` (void)  
*Make a list of the bluetooth address of paired device.*
- char `e_bt_remove_local_paired_device` (int j)  
*Remove a paired bluetooth device.*
- char `e_bt_set_event_filter` (char event)  
*Set event filter of module.*
- char `e_bt_get_event_filter` (void)  
*Get event filter of module if it is different than 0x02.*

## Variables

- unsigned char `e_bt_local_paired_device` [6 \*8]
- struct `BtDevice` `e_bt_present_device` [10]
- struct `BtEPuck` `e_bt_present_epuck` [10]
- char `local_bt_PIN` [4]

### 7.20.1 Detailed Description

Manage Bluetooth.

This module manage the Bluetooth device.

#### Author

Code: Michael Bonani  
Doc: Jonathan Besuchet

### 7.20.2 Function Documentation

7.20.2.1 char `e_bt_connect_epuck` ( void )

7.20.2.2 char `e_bt_establish_SPP_link` ( char \* address )

Try to connect to another bluetooth device.

**Parameters**

<i>address</i>	A pointer on the device address which you want to connect
----------------	---

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.3 void e\_bt\_exit\_tranparent\_mode ( void )**

Exit from the transparent mode.

**7.20.2.4 char e\_bt\_factory\_reset ( void )**

Factory reset of the bluetooth module.

**Warning**

use this function only if you are sure, your e-puck must be restarted, and renamed!!!

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.5 int e\_bt\_find\_epuck ( void )**

Try to find other e-puck.

This function make global inquiry and check which device are e-puck, and list them in globales tables.

**Returns**

number of e-puck found

**See also**

[e\\_bt\\_present\\_device](#), [e\\_bt\\_present\\_epuck](#)

**7.20.2.6 char e\_bt\_get\_event\_filter ( void )**

Get event filter of module if it is different than 0x02.

**Returns**

event filter 0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported

**7.20.2.7 char e\_bt\_get\_friendly\_name ( struct BtDevice \* device )**

To get the friendly name of a bluetooth device.

**Parameters**

<i>device</i>	A pointer on the device that you want the name
---------------	--

**Returns**

bluetooth error if one occur, 0 otherwise

**See also**

[BtDevice](#)

**7.20.2.8 int e\_bt\_inquiry ( struct BtDevice \* *device* )**

Research all the accessible bluetooth devices.

**Parameters**

<i>device</i>	A pointer to <a href="#">BtDevice</a> to store all the characteristics of each devices found
---------------	--

**Returns**

the number of device found

**See also**

[BtDevice](#), [e\\_bt\\_present\\_device](#)

**7.20.2.9 char e\_bt\_list\_local\_paired\_device ( void )**

Make a list of the bluetooth address of paired device.

**Returns**

The number of device found

**See also**

[e\\_bt\\_local\\_paired\\_device](#)

**7.20.2.10 char e\_bt\_read\_local\_name ( char \* *name* )**

Read the name of this e-puck's bluetooth module.

**Parameters**

<i>name</i>	A pointer to store the name
-------------	-----------------------------

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.11 char e\_bt\_read\_local\_pin\_number ( char \* PIN )**

Read the PIN number of this e-puck's bluetooth module.

**Parameters**

<i>PIN</i>	A pointer to store the PIN number
------------	-----------------------------------

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.12 char e\_bt\_release\_SPP\_link ( void )**

Unconnect from the current bluetooth device.

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.13 char e\_bt\_remove\_local\_paired\_device ( int j )**

Remove a paired bluetooth device.

**Parameters**

<i>j</i>	The number of the paired device to remove from the e_bt_local_paired_device array.
----------	--

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.14 int e\_bt\_reset ( void )**

Reset the bluetooth module.



**Returns**

The version number

**7.20.2.15** `char e_bt_send_SPP_data ( char * data, char datalength )`

Send data to the current bluetooth device.

**Warning**

send maximum 127 bytes if you are in non transparent mode

**Parameters**

<i>data</i>	The datas to send
<i>datalength</i>	The length of the datas to send

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.16** `char e_bt_set_event_filter ( char event )`

Set event filter of module.

**Warning**

this function can change the Bluetooth behaviours and than this library could not work anymore.

**Parameters**

<i>event</i>	0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported 0x03: No events generated and no UART break generated when device leaves transparent mode
--------------	---

**Returns**

bluetooth error if one occur, 0 otherwise

**7.20.2.17** `char e_bt_tranparent_mode ( void )`

Change to transparent mode.

**Returns**

bluetooth error if one occur, 0 otherwise

7.20.2.18 `char e_bt_write_local_name ( char * name )`

Write the name on this e-puck's bluetooth module.

## Parameters

<i>name</i>	A pointer to store the name
-------------	-----------------------------

## Returns

bluetooth error if one occur, 0 otherwise

## 7.20.2.19 char e\_bt\_write\_local\_pin\_number ( char \* PIN )

Write the PIN number on this e-puck's bluetooth module.

## Parameters

<i>PIN</i>	A pointer to store the PIN number
------------	-----------------------------------

## Returns

bluetooth error if one occur, 0 otherwise

## 7.20.3 Variable Documentation

## 7.20.3.1 unsigned char e\_bt\_local\_paired\_device[6 \*8]

## 7.20.3.2 struct BtDevice e\_bt\_present\_device[10]

An extern array containing all the bluetooth device detected. It's carried out by the fonction e\_bt\_find\_epuck

## See also

[e\\_bt\\_find\\_epuck](#)

## 7.20.3.3 struct BtEPuck e\_bt\_present\_epuck[10]

An extern array containing all the e-puck detected. It's carried out by the fonction e\_bt\_find\_epuck

## See also

[e\\_bt\\_find\\_epuck](#)

## 7.20.3.4 char local\_bt\_PIN[4]

## 7.21 bluetooth/e\_bluetooth.h File Reference

Manage Bluetooth.

## Data Structures

- struct [BtDevice](#)  
*general struct for bluetooth device*
- struct [BtEPuck](#)  
*general struct for other e-puck*

## Functions

- int [e\\_bt\\_find\\_epuck](#) (void)  
*Try to find other e-puck.*
- char [e\\_bt\\_connect\\_epuck](#) (void)
- char [e\\_bt\\_read\\_local\\_pin\\_number](#) (char \*PIN)  
*Read the PIN number of this e-puck's bluetooth module.*
- char [e\\_bt\\_read\\_local\\_name](#) (char \*name)  
*Read the name of this e-puck's bluetooth module.*
- char [e\\_bt\\_write\\_local\\_pin\\_number](#) (char \*PIN)  
*Write the PIN number on this e-puck's bluetooth module.*
- char [e\\_bt\\_write\\_local\\_name](#) (char \*name)  
*Write the name on this e-puck's bluetooth module.*
- char [e\\_bt\\_establish\\_SPP\\_link](#) (char \*address)  
*Try to connect to another bluetooth device.*
- char [e\\_bt\\_release\\_SPP\\_link](#) (void)  
*Disconnect from the current bluetooth device.*
- char [e\\_bt\\_send\\_SPP\\_data](#) (char \*data, char datalenght)  
*Send data to the current bluetooth device.*
- char [e\\_bt\\_tranparent\\_mode](#) (void)  
*Change to transparent mode.*
- void [e\\_bt\\_exit\\_tranparent\\_mode](#) (void)  
*Exit from the transparent mode.*
- char [e\\_bt\\_list\\_local\\_paired\\_device](#) (void)  
*Make a list of the bluetooth address of paired device.*
- char [e\\_bt\\_remove\\_local\\_paired\\_device](#) (int)  
*Remove a paired bluetooth device.*
- int [e\\_bt\\_inquiry](#) (struct [BtDevice](#) \*device)  
*Research all the accessible bluetooth devices.*
- char [e\\_bt\\_get\\_friendly\\_name](#) (struct [BtDevice](#) \*device)  
*To get the friendly name of a bluetooth device.*
- char [e\\_bt\\_set\\_event\\_filter](#) (char event)  
*Set event filter of module.*
- char [e\\_bt\\_get\\_event\\_filter](#) (void)  
*Get event filter of module if it is different than 0x02.*
- int [e\\_bt\\_reset](#) (void)  
*Reset the bluetooth module.*
- char [e\\_bt\\_factory\\_reset](#) (void)  
*Factory reset of the bluetooth module.*

## Variables

- unsigned char [e\\_bt\\_local\\_paired\\_device](#) [6 \*8]
- struct [BtDevice](#) [e\\_bt\\_present\\_device](#) [10]
- struct [BtEPuck](#) [e\\_bt\\_present\\_epuck](#) [10]

## 7.21.1 Detailed Description

Manage Bluetooth.

This module manage the bluetooth device.

### Author

Code: Michael Bonani

Doc: Jonathan Besuchet

## 7.21.2 Function Documentation

7.21.2.1 `char e_bt_connect_epuck ( void )`

7.21.2.2 `char e_bt_establish_SPP_link ( char * address )`

Try to connect to another bluetooth device.

### Parameters

<i>address</i>	A pointer on the device address which you want to connect
----------------	---

### Returns

bluetooth error if one occur, 0 otherwise

7.21.2.3 `void e_bt_exit_transparent_mode ( void )`

Exit from the transparent mode.

7.21.2.4 `char e_bt_factory_reset ( void )`

Factory reset of the bluetooth module.

### Warning

use this function only if you are sure, your e-puck must be restarted, and renamed!!!

### Returns

bluetooth error if one occur, 0 otherwise

#### 7.21.2.5 int e\_bt\_find\_epuck ( void )

Try to find other e-puck.

This function make global inquiry and check which device are e-puck, and list them in globales tables.

##### Returns

number of e-puck found

##### See also

[e\\_bt\\_present\\_device](#), [e\\_bt\\_present\\_epuck](#)

#### 7.21.2.6 char e\_bt\_get\_event\_filter ( void )

Get event filter of module if it is different than 0x02.

##### Returns

event filter 0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported

#### 7.21.2.7 char e\_bt\_get\_friendly\_name ( struct BtDevice \* device )

To get the friendly name of a bluetooth device.

##### Parameters

<i>device</i>	A pointer on the device that you want the name
---------------	--

##### Returns

bluetooth error if one occur, 0 otherwise

##### See also

[BtDevice](#)

#### 7.21.2.8 int e\_bt\_inquiry ( struct BtDevice \* device )

Research all the accessible bluetooth devices.

##### Parameters

<i>device</i>	A pointer to <a href="#">BtDevice</a> to store all the characteristics of each devices found
---------------	--

**Returns**

the number of device found

**See also**

[BtDevice](#), [e\\_bt\\_present\\_device](#)

**7.21.2.9** `char e_bt_list_local_paired_device ( void )`

Make a list of the bluetooth address of paired device.

**Returns**

The number of device found

**See also**

[e\\_bt\\_local\\_paired\\_device](#)

**7.21.2.10** `char e_bt_read_local_name ( char * name )`

Read the name of this e-puck's bluetooth module.

**Parameters**

<i>name</i>	A pointer to store the name
-------------	-----------------------------

**Returns**

bluetooth error if one occur, 0 otherwise

**7.21.2.11** `char e_bt_read_local_pin_number ( char * PIN )`

Read the PIN number of this e-puck's bluetooth module.

**Parameters**

<i>PIN</i>	A pointer to store the PIN number
------------	-----------------------------------

**Returns**

bluetooth error if one occur, 0 otherwise

#### 7.21.2.12 `char e_bt_release_SPP_link ( void )`

Unconnect from the current bluetooth device.

##### Returns

bluetooth error if one occur, 0 otherwise

#### 7.21.2.13 `char e_bt_remove_local_paired_device ( int j )`

Remove a paired bluetooth device.

##### Parameters

<i>j</i>	The number of the paired device to remove from the <code>e_bt_local_paired_device</code> array.
----------	---

##### Returns

bluetooth error if one occur, 0 otherwise

#### 7.21.2.14 `int e_bt_reset ( void )`

Reset the bluetooth module.

##### Returns

The version number

#### 7.21.2.15 `char e_bt_send_SPP_data ( char * data, char datalength )`

Send data to the current bluetooth device.

##### Warning

send maximum 127 bytes if you are in non transparent mode

##### Parameters

<i>data</i>	The datas to send
<i>datalength</i>	The length of the datas to send

##### Returns

bluetooth error if one occur, 0 otherwise



#### 7.21.2.16 char e\_bt\_set\_event\_filter ( char event )

Set event filter of module.

##### Warning

this function can change the Bluetooth behaviours and than this library could not work anymore.

##### Parameters

<i>event</i>	0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported 0x03: No events generated and no UART break generated when device leaves transparent mode
--------------	---

##### Returns

bluetooth error if one occur, 0 otherwise

#### 7.21.2.17 char e\_bt\_tranparent\_mode ( void )

Change to transparent mode.

##### Returns

bluetooth error if one occur, 0 otherwise

#### 7.21.2.18 char e\_bt\_write\_local\_name ( char \* name )

Write the name on this e-puck's bluetooth module.

##### Parameters

<i>name</i>	A pointer to store the name
-------------	-----------------------------

##### Returns

bluetooth error if one occur, 0 otherwise

#### 7.21.2.19 char e\_bt\_write\_local\_pin\_number ( char \* PIN )

Write the PIN number on this e-puck's bluetooth module.

##### Parameters

<i>PIN</i>	A pointer to store the PIN number
------------	-----------------------------------

**Returns**

bluetooth error if one occur, 0 otherwise

**7.21.3 Variable Documentation**

7.21.3.1 unsigned char `e_bt_local_paired_device[6 * 8]`

7.21.3.2 struct `BtDevice e_bt_present_device[10]`

An extern array containing all the bluetooth device detected. It's carried out by the fonction `e_bt_find_epuck`

**See also**

[e\\_bt\\_find\\_epuck](#)

7.21.3.3 struct `BtEPuck e_bt_present_epuck[10]`

An extern array containing all the e-puck detected. It's carried out by the fonction `e_bt_find_epuck`

**See also**

[e\\_bt\\_find\\_epuck](#)

**7.22 camera/fast\_2\_timer/e\_calc\_po3030k.c File Reference**

Calculate the timing for the camera (two timers)

```
#include "e_poxxxx.h"
#include "e_po3030k.h"
#include "../motor_led/e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../motor_led/e_init_port.h"
```

**Macros**

- #define `ARRAY_ORIGINE_X` 210
- #define `ARRAY_ORIGINE_Y` 7
- #define `IRQ_PIX_LAT` 1

**Functions**

- int `e_po3030k_config_cam` (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- int `e_po3030k_get_bytes_per_pixel` (int color\_mode)

## 7.22.1 Detailed Description

Calculate the timing for the camera (two timers)

### Author

Philippe Rétornaz

## 7.22.2 Macro Definition Documentation

7.22.2.1 `#define ARRAY_ORIGINE_X 210`

7.22.2.2 `#define ARRAY_ORIGINE_Y 7`

7.22.2.3 `#define IRQ_PIX_LAT 1`

## 7.22.3 Function Documentation

7.22.3.1 `int e_po3030k_config_cam ( unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact_width, unsigned int zoom_fact_height, int color_mode )`

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

### Parameters

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the insterest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

### Returns

Zero if the settings are correct, non-zero if an error occur

### See also

[e\\_po3030k\\_write\\_cam\\_registers](#)

### 7.22.3.2 `int e_po3030k_get_bytes_per_pixel ( int color_mode )`

Return the number of bytes per pixel in the given color mode

#### Parameters

<code>color_mode</code>	The given color mode
-------------------------	----------------------

#### Returns

The number of bytes per pixel in the given color mode

## 7.23 `camera/fast_2_timer/e_calc_po6030k.c` File Reference

Calculate the timing for the camera.

```
#include "e_po6030k.h"
#include "../motor_led/e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../motor_led/e_init_port.h"
```

#### Macros

- `#define ARRAY_ORIGINE_X 80`
- `#define ARRAY_ORIGINE_Y 8`
- `#define IRQ_PIX_LAT 1`

#### Functions

- `int e_po6030k_config_cam` (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- `int e_po6030k_get_bytes_per_pixel` (int color\_mode)

### 7.23.1 Detailed Description

Calculate the timing for the camera.

#### Author

Philippe R?ornaz

## 7.23.2 Macro Definition Documentation

7.23.2.1 `#define ARRAY_ORIGINE_X 80`

7.23.2.2 `#define ARRAY_ORIGINE_Y 8`

7.23.2.3 `#define IRQ_PIX_LAT 1`

## 7.23.3 Function Documentation

7.23.3.1 `int e_po6030k_config_cam ( unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact_width, unsigned int zoom_fact_height, int color_mode )`

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

### Parameters

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

### Returns

Zero if the settings are correct, non-zero if an error occur

7.23.3.2 `int e_po6030k_get_bytes_per_pixel ( int color_mode )`

Return the number of bytes per pixel in the given color mode

### Parameters

<i>color_mode</i>	The given color mode
-------------------	----------------------

### Returns

The number of bytes per pixel in the given color mode

## 7.24 camera/fast\_2\_timer/e\_calc\_po8030d.c File Reference

Calculate the timing for the camera.

```
#include "e_po8030d.h"
#include "../motor_led/e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../motor_led/e_init_port.h"
```

### Macros

- #define [ARRAY\\_ORIGINE\\_X](#) 68
- #define [ARRAY\\_ORIGINE\\_Y](#) 4
- #define [IRQ\\_PIX\\_LAT](#) 1

### Functions

- int [e\\_po8030d\\_config\\_cam](#) (unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode*)
- int [e\\_po8030d\\_get\\_bytes\\_per\\_pixel](#) (int *color\_mode*)

#### 7.24.1 Detailed Description

Calculate the timing for the camera.

#### Author

Stefano Morgani

#### 7.24.2 Macro Definition Documentation

7.24.2.1 #define [ARRAY\\_ORIGINE\\_X](#) 68

7.24.2.2 #define [ARRAY\\_ORIGINE\\_Y](#) 4

7.24.2.3 #define [IRQ\\_PIX\\_LAT](#) 1

#### 7.24.3 Function Documentation

7.24.3.1 int [e\\_po8030d\\_config\\_cam](#) ( unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode* )

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

#### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

**Parameters**

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

**Returns**

Zero if the settings are correct, non-zero if an error occur

## 7.24.3.2 int e\_po8030d\_get\_bytes\_per\_pixel ( int color\_mode )

Return the number of bytes per pixel in the given color mode

**Parameters**

<i>color_mode</i>	The given color mode
-------------------	----------------------

**Returns**

The number of bytes per pixel in the given color mode

**7.25 camera/fast\_2\_timer/e\_common.c File Reference**

```
#include "e_poxxxx.h"
#include "e_po3030k.h"
#include "e_po6030k.h"
#include "e_po8030d.h"
#include "../motor_led/e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../motor_led/e_init_port.h"
```

**Macros**

- #define `DEVICE_ID` 0xDC

**Functions**

- int `e_poxxxx_config_cam` (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- void `e_poxxxx_set_mirror` (int vertical, int horizontal)

- void `e_poxxxx_write_cam_registers` (void)
- int `e_poxxxx_init_cam` (void)
- static unsigned `readee` (unsigned page, unsigned addr)
- int `e_poxxxx_get_orientation` (void)
- void `e_poxxxx_set_awb_ae` (int awb, int ae)
- void `e_poxxxx_set_rgb_gain` (unsigned char r, unsigned char g, unsigned char b)
- void `e_poxxxx_set_exposure` (unsigned long exp)
- unsigned int `getCameraVersion` ()

## Variables

- static unsigned int `camera_version`

## 7.25.1 Macro Definition Documentation

7.25.1.1 `#define DEVICE_ID 0xDC`

## 7.25.2 Function Documentation

7.25.2.1 int `e_poxxxx_config_cam` ( unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode* )

7.25.2.2 int `e_poxxxx_get_orientation` ( void )

Return the camera orientation

### Returns

0: vertical 480x640, 1: horizontal 640x480, horizontal inverted, -1: unknown

7.25.2.3 int `e_poxxxx_init_cam` ( void )

Initialize the camera, return the version in hexa, 0x3030 or 0x6030

7.25.2.4 void `e_poxxxx_set_awb_ae` ( int *awb*, int *ae* )

Enable/Disable AWB and AE

### Parameters

<i>awb</i>	1 means AWB enabled, 0 means disabled
<i>ae</i>	1 means AE enabled, 0 means disabled



7.25.2.5 void e\_poxxxx\_set\_exposure ( unsigned long *exp* )

Set exposure time

Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

Warning

Only meaningful if AE is disabled

7.25.2.6 void e\_poxxxx\_set\_mirror ( int *vertical*, int *horizontal* )

7.25.2.7 void e\_poxxxx\_set\_rgb\_gain ( unsigned char *r*, unsigned char *g*, unsigned char *b* )

Set the gains of the camera

Parameters

<i>red</i>	The red pixels' gain (0..255)
<i>green</i>	The green pixels' gain (0..255)
<i>blue</i>	The blue pixels' gain (0..255)

Warning

Only meaningful if AWB is disabled

7.25.2.8 void e\_poxxxx\_write\_cam\_registers ( void )

7.25.2.9 unsigned int getCameraVersion ( )

7.25.2.10 static unsigned readee ( unsigned *page*, unsigned *addr* ) [static]

## 7.25.3 Variable Documentation

7.25.3.1 unsigned int camera\_version [static]

## 7.26 camera/fast\_2\_timer/e\_po3030k.h File Reference

PO3030k library header (two timers)

```
#include "e_poxxxx.h"
```

## Macros

- #define `PO3030K_FULL` 1
- #define `MODE_VGA` 0x44
- #define `MODE_QVGA` 0x11
- #define `MODE_QQVGA` 0x33
- #define `SPEED_2` 0x00
- #define `SPEED_2_3` 0x10
- #define `SPEED_4` 0x20
- #define `SPEED_8` 0x30
- #define `SPEED_16` 0x40
- #define `SPEED_32` 0x50
- #define `SPEED_64` 0x60
- #define `SPEED_128` 0x70

## Functions

- int `e_po3030k_config_cam` (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- int `e_po3030k_get_bytes_per_pixel` (int color\_mode)
- void `e_po3030k_init_cam` (void)
- void `e_po3030k_write_cam_registers` (void)
- int `e_po3030k_set_color_mode` (int mode)
- int `e_po3030k_set_sampling_mode` (int mode)
- int `e_po3030k_set_speed` (int mode)
- int `e_po3030k_set_wx` (unsigned int start, unsigned int stop)
- int `e_po3030k_set_wy` (unsigned int start, unsigned int stop)
- int `e_po3030k_set_vsync` (unsigned int start, unsigned int stop, unsigned int col)
- void `e_po3030k_set_mirror` (int vertical, int horizontal)
- void `e_po3030k_read_cam_registers` (void)
- int `e_po3030k_set_register` (unsigned char adr, unsigned char value)
- int `e_po3030k_get_register` (unsigned char adr, unsigned char \*value)
- void `e_po3030k_set_bias` (unsigned char pixbias, unsigned char opbias)
- void `e_po3030k_set_integr_time` (unsigned long time)
- void `e_po3030k_set_adc_offset` (unsigned char offset)
- void `e_po3030k_set_sepia` (int status)
- void `e_po3030k_set_lens_gain` (unsigned char red, unsigned char green, unsigned char blue)
- void `e_po3030k_set_edge_prop` (unsigned char gain, unsigned char tresh)
- void `e_po3030k_set_gamma_coef` (unsigned char array[12], char color)
- void `e_po3030k_write_gamma_coef` (void)
- int `e_po3030k_sync_register_array` (unsigned char start, unsigned char stop)
- void `e_po3030k_set_color_matrix` (unsigned char array[3 \* 3])
- void `e_po3030k_set_cb_cr_gain` (unsigned char cg11c, unsigned char cg22c)
- void `e_po3030k_set_brigh_contr` (unsigned char bright, unsigned char contrast)
- void `e_po3030k_set_sepia_tone` (unsigned char cb, unsigned char cr)
- void `e_po3030k_set_ww` (unsigned char ww)
- void `e_po3030k_set_awb_ae_tol` (unsigned char awbm, unsigned char aem)
- void `e_po3030k_set_ae_speed` (unsigned char b, unsigned char d)
- void `e_po3030k_set_exposure` (long t)
- void `e_po3030k_set_ref_exposure` (unsigned char exp)
- void `e_po3030k_set_max_min_exp` (unsigned int min, unsigned int max)
- void `e_po3030k_set_max_min_awb` (unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob)
- int `e_po3030k_set_weight_win` (unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2)

- void `e_po3030k_set_awb_ae` (int awb, int ae)
- int `e_po3030k_set_color_gain` (unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)
- void `e_po3030k_set_flicker_mode` (int manual)
- void `e_po3030k_set_flicker_detection` (int hz50, int hz60)
- int `e_po3030k_set_flicker_man_set` (int hz50, int hz60, int fdm, int fk, int tol)

### 7.26.1 Detailed Description

PO3030k library header (two timers)

Author

Philippe Rétornaz

### 7.26.2 Macro Definition Documentation

7.26.2.1 `#define MODE_QQVGA 0x33`

7.26.2.2 `#define MODE_QVGA 0x11`

7.26.2.3 `#define MODE_VGA 0x44`

7.26.2.4 `#define PO3030K_FULL 1`

If you set this at 0, you save about 168 bytes of memory But you loose all advanced camera functions

7.26.2.5 `#define SPEED_128 0x70`

7.26.2.6 `#define SPEED_16 0x40`

7.26.2.7 `#define SPEED_2 0x00`

7.26.2.8 `#define SPEED_2_3 0x10`

7.26.2.9 `#define SPEED_32 0x50`

7.26.2.10 `#define SPEED_4 0x20`

7.26.2.11 `#define SPEED_64 0x60`

7.26.2.12 `#define SPEED_8 0x30`

### 7.26.3 Function Documentation

7.26.3.1 int `e_po3030k_config_cam` ( unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode* )

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

#### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

**Parameters**

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

**Returns**

Zero if the settings are correct, non-zero if an error occur

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

**Warning**

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2.

**Parameters**

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

**Returns**

Zero if the settings are correct, non-zero if an error occur

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

### 7.26.3.2 int e\_po3030k\_get\_bytes\_per\_pixel ( int *color\_mode* )

Return the number of bytes per pixel in the given color mode

## Parameters

<i>color_mode</i>	The given color mode
-------------------	----------------------

## Returns

The number of bytes per pixel in the given color mode

### 7.26.3.3 int e\_po3030k\_get\_register ( unsigned char *adr*, unsigned char \* *value* )

Get the register *adr* value

## Parameters

<i>adr</i>	The address
<i>value</i>	The pointer to the value to write to

## Returns

Zero if register found, non-zero if not found

## Warning

This function is sub-optimal, if you use it heavily add an internal function to register.c

## See also

[e\\_po3030k\\_set\\_register](#)

### 7.26.3.4 void e\_po3030k\_init\_cam ( void )

Initialize the camera, must be called before any other function

### 7.26.3.5 void e\_po3030k\_read\_cam\_registers ( void )

Read the camera register

## See also

[e\\_po3030k\\_write\\_cam\\_registers](#)

### 7.26.3.6 void e\_po3030k\_set\_adc\_offset ( unsigned char *offset* )

Set the Analog to Digital Converter offset

## Parameters

<i>offset</i>	The offset
---------------	------------

## See also

Datasheet p.28

7.26.3.7 void e\_po3030k\_set\_ae\_speed ( unsigned char *b*, unsigned char *d* )

Set AE speed

## Parameters

<i>b</i>	AE speed factor when exposure time is decreasing ( 4 lower bits only )
<i>d</i>	AE speed factor when exposure time is increasing ( 4 lower bits only )

## See also

Datasheet p.44

7.26.3.8 void e\_po3030k\_set\_awb\_ae ( int *awb*, int *ae* )

Enable/Disable AWB and AE

## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

## See also

Datasheet p. 50

7.26.3.9 void e\_po3030k\_set\_awb\_ae\_tol ( unsigned char *awbm*, unsigned char *aem* )

Set AWB/AE tolerance margin

## Parameters

<i>awbm</i>	AWV Margin ( 4 lower bits only )
<i>aem</i>	AW Margin ( 4 lower bits only )

## See also

Datasheet p.44

7.26.3.10 void e\_po3030k\_set\_bias ( unsigned char *pixbias*, unsigned char *opbias* )

Set the Pixel and amplificator bias Increasing the bias produce better image quality, but increase camera's power consumption

## Parameters

<i>pixbias</i>	The pixel bias
<i>opbias</i>	The Amplificator bias

## See also

Datasheet p.22

7.26.3.11 void e\_po3030k\_set\_brigh\_contr ( unsigned char *bright*, unsigned char *contrast* )

Set the Brightness & Contrast

## Parameters

<i>bright</i>	The Brightness ( signed 1+7bits fixed point format )
<i>contrast</i>	The Contrast

## See also

Datasheet p. 40

7.26.3.12 void e\_po3030k\_set\_cb\_cr\_gain ( unsigned char *cg11c*, unsigned char *cg22c* )

Set The color gain ( Cb/Cr )

## Parameters

<i>cg11c</i>	Cb gain ( Sign[7]   Integer[6:5]   fractional[4:0] )
<i>cg22c</i>	Cr gain ( Sign[7]   Integer[6:5]   fractional[4:0] )

## See also

Datasheet p. 40

7.26.3.13 `int e_po3030k_set_color_gain ( unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue )`

Set the gains of the camera

#### Parameters

<i>global</i>	The global gain $\in \{0, 79\}$
<i>red</i>	The red pixel's gain (fixed point [2:6] format)
<i>green1</i>	The green pixel near read one gain ([2:6] format)
<i>green2</i>	The green pixel near blue one gain ([2:6] format)
<i>blue</i>	The blue pixel's gain ([2:6] format)

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p.23-24

7.26.3.14 `void e_po3030k_set_color_matrix ( unsigned char array[3*3] )`

7.26.3.15 `int e_po3030k_set_color_mode ( int mode )`

Set the camera color mode

#### Warning

This is an internal function, use `e_po3030k_config_cam`

#### Parameters

<i>mode</i>	The color mode
-------------	----------------

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p. 31, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

7.26.3.16 `void e_po3030k_set_edge_prop ( unsigned char gain, unsigned char tresh )`

Set Edge properties



## Parameters

<i>gain</i>	Edge gain & moire factor (fixed point [2:3] format)
<i>tresh</i>	Edge Enhancement threshold

## See also

Datasheet p.36

## 7.26.3.17 void e\_po3030k\_set\_exposure ( long t )

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

## Warning

Only writable if AE is disabled

## See also

Datasheet p.45

## 7.26.3.18 void e\_po3030k\_set\_flicker\_detection ( int hz50, int hz60 )

Set the 50/60Hz flicker detection

## Parameters

<i>hz50</i>	Non-zero mean 50Hz flicker detection enabled (default disabled)
<i>hz60</i>	Non-zero mean 60Hz flicker detection enabled (default disabled)

## Warning

If Automatic mode is enabled and both 50Hz and 60Hz are disabled, camera will enable both. By default, the camera automatically detect 50 and 60Hz flicker.

## See also

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

## 7.26.3.19 int e\_po3030k\_set\_flicker\_man\_set ( int hz50, int hz60, int fdm, int fk, int tol )

Set the camera's manual flicker's detection setting

**Parameters**

<i>hz50</i>	The Hz for the 50Hz detection
<i>hz60</i>	The Hz for the 60Hz detection
<i>fdm</i>	Flicker duration mode
<i>fk</i>	Flicker count step
<i>tol</i>	Flicker tolerance

**Warning**

You must have set the mode ( image size, color ) before calling this function

**Returns**

Non-zero if an error occur, 0 if OK

**See also**

Datasheet p.29-30 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#)

**7.26.3.20 void e\_po3030k\_set\_flicker\_mode ( int *manual* )**

Set flicker detection mode

**Parameters**

<i>manual</i>	Non-zero mean manual mode is enabled ( default automatic mode enabled )
---------------	---

**See also**

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

**7.26.3.21 void e\_po3030k\_set\_gamma\_coef ( unsigned char *array*[12], char *color* )**

Set gamma coefficient

**Warning**

This feature need extra care from the user

**Parameters**

<i>array</i>	Gamma coefficient array
<i>color</i>	First two bytes : - 0b01 => Green <ul style="list-style-type: none"> <li>• 0b00 =&gt; Red</li> <li>• else =&gt; Blue</li> </ul>

## See also

Datasheet p. 38, 50, 57-58 and e\_po3030k\_WriteGammaCoef

7.26.3.22 void e\_po3030k\_set\_integr\_time ( unsigned long *time* )

Set the pixel intergration time This is counted in line-time interval. See dataseet p.25 for more information

## Parameters

<i>time</i>	The integration time ( fixed point [14:6] format )
-------------	--

## See also

Datasheet p.25

7.26.3.23 void e\_po3030k\_set\_lens\_gain ( unsigned char *red*, unsigned char *green*, unsigned char *blue* )

Set lens shading gain

## Parameters

<i>red</i>	Lens gain for red pixel $\in \{0, 15\}$
<i>green</i>	Lens gain for green pixel $\in \{0, 15\}$
<i>blue</i>	Lens gain for blue pixel $\in \{0, 15\}$

## See also

Datasheet p.36

7.26.3.24 void e\_po3030k\_set\_max\_min\_awb ( unsigned char *minb*, unsigned char *maxb*, unsigned char *minr*, unsigned char *maxr*, unsigned char *ratior*, unsigned char *ratiob* )

Set the minimum and maximum red and blue gain in AWB mode

## Parameters

<i>minb</i>	The minimum blue gain
<i>maxb</i>	The maximum blue gain
<i>minr</i>	The minimum red gain
<i>maxr</i>	The maximum red gain
<i>ratior</i>	The red gain ratio
<i>ratiob</i>	The blue gain ratio

**See also**

Datasheet p. 47-48

### 7.26.3.25 void e\_po3030k\_set\_max\_min\_exp ( unsigned int *max*, unsigned int *min* )

Set the minimum and maximum exposure time in AE mode

**Parameters**

<i>min</i>	The minimum exposure time
<i>max</i>	The maximum exposure time

**See also**

Datasheet p.46-47

### 7.26.3.26 void e\_po3030k\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

**Parameters**

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

**See also**

Datasheet p.27

### 7.26.3.27 void e\_po3030k\_set\_ref\_exposure ( unsigned char *exp* )

Set the reference exposure. The average brightness which the AE should have

**Parameters**

<i>exp</i>	The target exposure level
------------	---------------------------

**See also**

Datasheet p.45

### 7.26.3.28 int e\_po3030k\_set\_register ( unsigned char *adr*, unsigned char *value* )

Set the register *adr* to value *value*

**Parameters**

<i>adr</i>	The address
<i>value</i>	The value

**Returns**

Zero if register found, non-zero if not found

**Warning**

This function is sub-optimal, if you use it heavily add an internal function to register.c

**See also**

[e\\_po3030k\\_get\\_register](#)

**7.26.3.29 int e\_po3030k\_set\_sampling\_mode ( int mode )**

Set the camera sampling mode

**Warning**

This is an internal function, use [e\\_po3030k\\_config\\_cam](#)

**Parameters**

<i>mode</i>	The given sampling mode
-------------	-------------------------

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 28 and [e\\_po3030k\\_config\\_cam](#)

**7.26.3.30 void e\_po3030k\_set\_sepia ( int status )**

Enable/Disable Sepia color

**Parameters**

<i>status</i>	Set <i>status</i> to 1 to enable, 0 to disable
---------------	--

**See also**

Datasheet p.34 and 74

**7.26.3.31 void e\_po3030k\_set\_sepia\_tone ( unsigned char *cb*, unsigned char *cr* )**

Set The color tone at sepia color condition

**Parameters**

<i>cb</i>	Cb tone
<i>cr</i>	Cr tone

**See also**

[e\\_po3030k\\_set\\_sepia](#) and Datasheet p. 41 and 74

**7.26.3.32 int e\_po3030k\_set\_speed ( int *mode* )**

Set the camera speed

**Warning**

This is an internal function, use [e\\_po3030k\\_config\\_cam](#)

**Parameters**

<i>mode</i>	The given speed
-------------	-----------------

**Returns**

Zero if OK, non-zero if unknow mode

**See also**

Datasheet p. 26 and [e\\_po3030k\\_config\\_cam](#)

**7.26.3.33 int e\_po3030k\_set\_vsync ( unsigned int *start*, unsigned int *stop*, unsigned int *col* )**

Set the camera window VSYNC coordinate

**Warning**

This is an internal function, use [e\\_po3030k\\_ConfigCam](#)

## Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row
<i>col</i>	The start/stop column

## Returns

Zero if OK, non-zero if an error occur

## See also

Datasheet p.42-43, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

7.26.3.34 int e\_po3030k\_set\_weight\_win ( unsigned int *x1*, unsigned int *x2*, unsigned int *y1*, unsigned int *y2* )

Set the Weighting Window coordinate

## Parameters

<i>x1</i>	The X1 coordinate $\in \{211, x2\}$
<i>x2</i>	The X2 coordinate $\in \{x1 + 1, 423\}$
<i>y1</i>	The Y1 coordinate $\in \{160, y2\}$
<i>y2</i>	The Y2 coordinate $\in \{y1 + 1, 319\}$

## Returns

Zero if OK, non-zero if an error occur

## See also

Datasheet p. 49

7.26.3.35 void e\_po3030k\_set\_ww ( unsigned char *ww* )

Set the Center weight (Back Light compensation) Control parameter

## Parameters

<i>ww</i>	Center weight ( 4 lower bits only )
-----------	-------------------------------------

## See also

Datasheet p.44

### 7.26.3.36 int e\_po3030k\_set\_wx ( unsigned int *start*, unsigned int *stop* )

Set the camera window X coordinate

#### Warning

This is an internal function, use e\_po3030k\_ConfigCam

#### Parameters

<i>start</i>	The start column
<i>stop</i>	The stop column

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p.20-21, [e\\_po3030k\\_write\\_cam\\_registers](#), [e\\_po3030k\\_set\\_wy](#) and [e\\_po3030k\\_config\\_cam](#)

### 7.26.3.37 int e\_po3030k\_set\_wy ( unsigned int *start*, unsigned int *stop* )

Set the camera window Y coordinate

#### Warning

This is an internal function, use e\_po3030k\_ConfigCam

#### Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p.20-21, [e\\_po3030k\\_WriteCamRegisters](#), [e\\_po3030k\\_SetWX](#) and [e\\_po3030k\\_ConfigCam](#)

### 7.26.3.38 int e\_po3030k\_sync\_register\_array ( unsigned char *start*, unsigned char *stop* )

Write every known register between address start and stop (inclusively).



**Warning**

It's better to set the configuration with appropriate functions and then write all registers with `e_po3030k_↔  
WriteCamRegisters`

**Parameters**

<i>start</i>	The beginning address of the write
<i>stop</i>	The last write address

**Returns**

The number of register written

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.26.3.39 `void e_po3030k_write_cam_registers ( void )`

The Po3030k module keep in memory the state of each register the camera has. When you configure the camera, it only alter the internal register state, not the camera one. This function write the internal register state in the camera.

**See also**

[e\\_po3030k\\_read\\_cam\\_registers](#)

7.26.3.40 `void e_po3030k_write_gamma_coef ( void )`

This special function write directly the Gamma coefficient and Gamma color select into camera register.

**Warning**

This function need extra care from the user

**See also**

[e\\_po3030k\\_set\\_gamma\\_coef](#)

## 7.27 camera/slow\_3\_timer/e\_po3030k.h File Reference

PO3030k library header (three timers)

## Macros

- #define [PO3030K\\_FULL](#) 1
- #define [ARRAY\\_WIDTH](#) 640
- #define [ARRAY\\_HEIGHT](#) 480
- #define [GREY\\_SCALE\\_MODE](#) 0
- #define [RGB\\_565\\_MODE](#) 1
- #define [YUV\\_MODE](#) 2
- #define [MODE\\_VGA](#) 0x44
- #define [MODE\\_QVGA](#) 0x11
- #define [MODE\\_QQVGA](#) 0x33
- #define [SPEED\\_2](#) 0x00
- #define [SPEED\\_2\\_3](#) 0x10
- #define [SPEED\\_4](#) 0x20
- #define [SPEED\\_8](#) 0x30
- #define [SPEED\\_16](#) 0x40
- #define [SPEED\\_32](#) 0x50
- #define [SPEED\\_64](#) 0x60
- #define [SPEED\\_128](#) 0x70

## Functions

- int [e\\_po3030k\\_config\\_cam](#) (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- int [e\\_po3030k\\_get\\_bytes\\_per\\_pixel](#) (int color\_mode)
- void [e\\_po3030k\\_init\\_cam](#) (void)
- void [e\\_po3030k\\_write\\_cam\\_registers](#) (void)
- void [e\\_po3030k\\_launch\\_capture](#) (char \*buf)
- void [e\\_po3030k\\_apply\\_timer\\_config](#) (int pixel\_row, int pixel\_col, int bpp, int pbp, int bbl)
- int [e\\_po3030k\\_is\\_img\\_ready](#) (void)
- int [e\\_po3030k\\_set\\_color\\_mode](#) (int mode)
- int [e\\_po3030k\\_set\\_sampling\\_mode](#) (int mode)
- int [e\\_po3030k\\_set\\_speed](#) (int mode)
- int [e\\_po3030k\\_set\\_wx](#) (unsigned int start, unsigned int stop)
- int [e\\_po3030k\\_set\\_wy](#) (unsigned int start, unsigned int stop)
- int [e\\_po3030k\\_set\\_vsync](#) (unsigned int start, unsigned int stop, unsigned int col)
- void [e\\_po3030k\\_read\\_cam\\_registers](#) (void)
- int [e\\_po3030k\\_set\\_register](#) (unsigned char adr, unsigned char value)
- int [e\\_po3030k\\_get\\_register](#) (unsigned char adr, unsigned char \*value)
- void [e\\_po3030k\\_set\\_bias](#) (unsigned char pixbias, unsigned char opbias)
- void [e\\_po3030k\\_set\\_integr\\_time](#) (unsigned long time)
- void [e\\_po3030k\\_set\\_mirror](#) (int vertical, int horizontal)
- void [e\\_po3030k\\_set\\_adc\\_offset](#) (unsigned char offset)
- void [e\\_po3030k\\_set\\_sepia](#) (int status)
- void [e\\_po3030k\\_set\\_lens\\_gain](#) (unsigned char red, unsigned char green, unsigned char blue)
- void [e\\_po3030k\\_set\\_edge\\_prop](#) (unsigned char gain, unsigned char tresh)
- void [e\\_po3030k\\_set\\_gamma\\_coef](#) (unsigned char array[12], char color)
- void [e\\_po3030k\\_write\\_gamma\\_coef](#) (void)
- int [e\\_po3030k\\_sync\\_register\\_array](#) (unsigned char start, unsigned char stop)
- void [e\\_po3030k\\_set\\_color\\_matrix](#) (unsigned char array[3 \* 3])
- void [e\\_po3030k\\_set\\_cb\\_cr\\_gain](#) (unsigned char cg11c, unsigned char cg22c)
- void [e\\_po3030k\\_set\\_brigh\\_contr](#) (unsigned char bright, unsigned char contrast)
- void [e\\_po3030k\\_set\\_sepia\\_tone](#) (unsigned char cb, unsigned char cr)
- void [e\\_po3030k\\_set\\_ww](#) (unsigned char ww)

- void `e_po3030k_set_awb_ae_tol` (unsigned char awbm, unsigned char aem)
- void `e_po3030k_set_ae_speed` (unsigned char b, unsigned char d)
- void `e_po3030k_set_exposure` (long t)
- void `e_po3030k_set_ref_exposure` (unsigned char exp)
- void `e_po3030k_set_max_min_exp` (unsigned int min, unsigned int max)
- void `e_po3030k_set_max_min_awb` (unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob)
- int `e_po3030k_set_weight_win` (unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2)
- void `e_po3030k_set_awb_ae` (int awb, int ae)
- int `e_po3030k_set_color_gain` (unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)
- void `e_po3030k_set_flicker_mode` (int manual)
- void `e_po3030k_set_flicker_detection` (int hz50, int hz60)
- int `e_po3030k_set_flicker_man_set` (int hz50, int hz60, int fdm, int fk, int tol)

### 7.27.1 Detailed Description

PO3030k library header (three timers)

Author

Philippe Rétornaz

### 7.27.2 Macro Definition Documentation

7.27.2.1 `#define ARRAY_HEIGHT 480`

7.27.2.2 `#define ARRAY_WIDTH 640`

7.27.2.3 `#define GREY_SCALE_MODE 0`

7.27.2.4 `#define MODE_QQVGA 0x33`

7.27.2.5 `#define MODE_QVGA 0x11`

7.27.2.6 `#define MODE_VGA 0x44`

7.27.2.7 `#define PO3030K_FULL 1`

If you set this at 0, you save about 168 bytes of memory But you loose all advanced camera functions

7.27.2.8 `#define RGB_565_MODE 1`

7.27.2.9 `#define SPEED_128 0x70`

7.27.2.10 `#define SPEED_16 0x40`

7.27.2.11 `#define SPEED_2 0x00`

7.27.2.12 `#define SPEED_2_3 0x10`

7.27.2.13 `#define SPEED_32 0x50`

7.27.2.14 `#define SPEED_4 0x20`

7.27.2.15 `#define SPEED_64 0x60`

7.27.2.16 `#define SPEED_8 0x30`

7.27.2.17 `#define YUV_MODE 2`

### 7.27.3 Function Documentation

7.27.3.1 `void e_po3030k_apply_timer_config ( int pixel_row, int pixel_col, int bpp, int pbp, int bbl )`

Modify the interrupt configuration

#### Warning

This is an internal function, use `e_po3030k_config_cam`

#### Parameters

<i>pixel_row</i>	The number of row to take
<i>pixel_col</i>	The number of pixel to take each <i>pixel_row</i>
<i>bpp</i>	The number of byte per pixel
<i>pbp</i>	The number of pixel to ignore between each pixel
<i>bbl</i>	The number of row to ignore between each line

#### See also

[e\\_po3030k\\_get\\_bytes\\_per\\_pixel](#) and [e\\_po3030k\\_config\\_cam](#)

7.27.3.2 `int e_po3030k_config_cam ( unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact_width, unsigned int zoom_fact_height, int color_mode )`

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

**Warning**

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

**Parameters**

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

**Returns**

Zero if the settings are correct, non-zero if an error occur

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

**Warning**

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2.

**Parameters**

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

**Returns**

Zero if the settings are correct, non-zero if an error occur

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

### 7.27.3.3 int e\_po3030k\_get\_bytes\_per\_pixel ( int *color\_mode* )

Return the number of bytes per pixel in the given color mode

#### Parameters

<i>color_mode</i>	The given color mode
-------------------	----------------------

#### Returns

The number of bytes per pixel in the given color mode

### 7.27.3.4 int e\_po3030k\_get\_register ( unsigned char *adr*, unsigned char \* *value* )

Get the register *adr* value

#### Parameters

<i>adr</i>	The address
<i>value</i>	The pointer to the value to write to

#### Returns

Zero if register found, non-zero if not found

#### Warning

This function is sub-optimal, if you use it heavily add an internal function to register.c

#### See also

[e\\_po3030k\\_set\\_register](#)

### 7.27.3.5 void e\_po3030k\_init\_cam ( void )

Initialize the camera, must be called before any other function

### 7.27.3.6 int e\_po3030k\_is\_img\_ready ( void )

Check if the current capture is finished

#### Returns

Zero if the current capture is in progress, non-zero if the capture is done.

#### See also

[e\\_po3030k\\_launch\\_capture](#)

### 7.27.3.7 void e\_po3030k\_launch\_capture ( char \* *buf* )

Launch a capture in the *buf* buffer

## Parameters

<i>buf</i>	The buffer to write to
------------	------------------------

## See also

[e\\_po3030k\\_config\\_cam](#) and [e\\_po3030k\\_is\\_img\\_ready](#)

7.27.3.8 void e\_po3030k\_read\_cam\_registers ( void )

Read the camera register

## See also

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.27.3.9 void e\_po3030k\_set\_adc\_offset ( unsigned char *offset* )

Set the Analog to Digital Converter offset

## Parameters

<i>offset</i>	The offset
---------------	------------

## See also

Datasheet p.28

7.27.3.10 void e\_po3030k\_set\_ae\_speed ( unsigned char *b*, unsigned char *d* )

Set AE speed

## Parameters

<i>b</i>	AE speed factor when exposure time is decreasing ( 4 lower bits only )
<i>d</i>	AE speed factor when exposure time is increasing ( 4 lower bits only )

## See also

Datasheet p.44

7.27.3.11 void e\_po3030k\_set\_awb\_ae ( int *awb*, int *ae* )

Enable/Disable AWB and AE



## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

## See also

Datasheet p. 50

7.27.3.12 void e\_po3030k\_set\_awb\_ae\_tol ( unsigned char *awbm*, unsigned char *aem* )

Set AWB/AE tolerance margin

## Parameters

<i>awbm</i>	AWV Margin ( 4 lower bits only )
<i>aem</i>	AW Margin ( 4 lower bits only )

## See also

Datasheet p.44

7.27.3.13 void e\_po3030k\_set\_bias ( unsigned char *pixbias*, unsigned char *opbias* )

Set the Pixel and amplifacator bias Increasing the bias produce better image quality, but increase camera's power consumption

## Parameters

<i>pixbias</i>	The pixel bias
<i>opbias</i>	The Amplifacator bias

## See also

Datasheet p.22

7.27.3.14 void e\_po3030k\_set\_brigh\_contr ( unsigned char *bright*, unsigned char *contrast* )

Set the Brightness & Contrast

## Parameters

<i>bright</i>	The Brightness ( signed 1+7bits fixed point format )
<i>contrast</i>	The Contrast

**See also**

Datasheet p. 40

7.27.3.15 void e\_po3030k\_set\_cb\_cr\_gain ( unsigned char *cg11c*, unsigned char *cg22c* )

Set The color gain ( Cb/Cr )

**Parameters**

<i>cg11c</i>	Cb gain ( Sign[7]   Integer[6:5]   fractional[4:0] )
<i>cg22c</i>	Cr gain ( Sign[7]   Integer[6:5]   fractional[4:0] )

**See also**

Datasheet p. 40

7.27.3.16 int e\_po3030k\_set\_color\_gain ( unsigned char *global*, unsigned char *red*, unsigned char *green1*, unsigned char *green2*, unsigned char *blue* )

Set the gains of the camera

**Parameters**

<i>global</i>	The global gain $\in \{0, 79\}$
<i>red</i>	The red pixel's gain (fixed point [2:6] format)
<i>green1</i>	The green pixel near read one gain ([2:6] format)
<i>green2</i>	The green pixel near blue one gain ([2:6] format)
<i>blue</i>	The blue pixel's gain ([2:6] format)

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.23-24

7.27.3.17 void e\_po3030k\_set\_color\_matrix ( unsigned char *array*[3 \*3] )

7.27.3.18 int e\_po3030k\_set\_color\_mode ( int *mode* )

Set the camera color mode

**Warning**

This is an internal function, use e\_po3030k\_config\_cam

## Parameters

<i>mode</i>	The color mode
-------------	----------------

## Returns

Zero if OK, non-zero if an error occur

## See also

Datasheet p. 31, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

7.27.3.19 void `e_po3030k_set_edge_prop ( unsigned char gain, unsigned char tresh )`

Set Edge properties

## Parameters

<i>gain</i>	Edge gain & moire factor (fixed point [2:3] format)
<i>tresh</i>	Edge Enhancement threshold

## See also

Datasheet p.36

7.27.3.20 void `e_po3030k_set_exposure ( long t )`

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

## Warning

Only writable if AE is disabled

## See also

Datasheet p.45

7.27.3.21 void `e_po3030k_set_flicker_detection ( int hz50, int hz60 )`

Set the 50/60Hz flicker detection

## Parameters

<i>hz50</i>	Non-zero mean 50Hz flicker detection enabled (default disabled)
<i>hz60</i>	Non-zero mean 60Hz flicker detection enabled (default disabled)

## Warning

If Automatic mode is enabled and both 50Hz and 60Hz are disabled, camera will enable both. By default, the camera automatically detect 50 and 60Hz flicker.

## See also

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

7.27.3.22 `int e_po3030k_set_flicker_man_set ( int hz50, int hz60, int fdm, int fk, int tol )`

Set the camera's manual flicker's detection setting

## Parameters

<i>hz50</i>	The Hz for the 50Hz detection
<i>hz60</i>	The Hz for the 60Hz detection
<i>fdm</i>	Flicker duration mode
<i>fk</i>	Flicker count step
<i>tol</i>	Flicker tolerance

## Warning

You must have set the mode ( image size, color ) before calling this function

## Returns

Non-zero if an error occur, 0 if OK

## See also

Datasheet p.29-30 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#)

7.27.3.23 `void e_po3030k_set_flicker_mode ( int manual )`

Set flicker detection mode

## Parameters

<i>manual</i>	Non-zero mean manual mode is enabled ( default automatic mode enabled )
---------------	---

## See also

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

7.27.3.24 void e\_po3030k\_set\_gamma\_coef ( unsigned char *array*[12], char *color* )

Set gamma coefficient

## Warning

This feature need extra care from the user

## Parameters

<i>array</i>	Gamma coefficient array
<i>color</i>	First two bytes : - 0b01 => Green <ul style="list-style-type: none"> <li>• 0b00 =&gt; Red</li> <li>• else =&gt; Blue</li> </ul>

## See also

Datasheet p. 38, 50, 57-58 and [e\\_po3030k\\_WriteGammaCoef](#)

7.27.3.25 void e\_po3030k\_set\_integr\_time ( unsigned long *time* )

Set the pixel intergration time This is counted in line-time interval. See dataset p.25 for more information

## Parameters

<i>time</i>	The integration time ( fixed point [14:6] format )
-------------	--

## See also

Datasheet p.25

7.27.3.26 void e\_po3030k\_set\_lens\_gain ( unsigned char *red*, unsigned char *green*, unsigned char *blue* )

Set lens shading gain

## Parameters

<i>red</i>	Lens gain for red pixel $\in \{0, 15\}$
<i>green</i>	Lens gain for green pixel $\in \{0, 15\}$
<i>blue</i>	Lens gain for blue pixel $\in \{0, 15\}$

## See also

Datasheet p.36

7.27.3.27 void e\_po3030k\_set\_max\_min\_awb ( unsigned char *minb*, unsigned char *maxb*, unsigned char *minr*, unsigned char *maxr*, unsigned char *ratior*, unsigned char *ratiob* )

Set the minimum and maximum red and blue gain in AWB mode

## Parameters

<i>minb</i>	The minimum blue gain
<i>maxb</i>	The maximum blue gain
<i>minr</i>	The minimum red gain
<i>maxr</i>	The maximum red gain
<i>ratior</i>	The red gain ratio
<i>ratiob</i>	The blue gain ratio

## See also

Datasheet p. 47-48

7.27.3.28 void e\_po3030k\_set\_max\_min\_exp ( unsigned int *max*, unsigned int *min* )

Set the minimum and maximum exposure time in AE mode

## Parameters

<i>min</i>	The minimum exposure time
<i>max</i>	The maximum exposure time

## See also

Datasheet p.46-47

7.27.3.29 void e\_po3030k\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

## Parameters

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

**See also**

Datasheet p.27

**7.27.3.30 void e\_po3030k\_set\_ref\_exposure ( unsigned char *exp* )**

Set the reference exposure. The average brightness which the AE should have

**Parameters**

<i>exp</i>	The target exposure level
------------	---------------------------

**See also**

Datasheet p.45

**7.27.3.31 int e\_po3030k\_set\_register ( unsigned char *adr*, unsigned char *value* )**

Set the register *adr* to value *value*

**Parameters**

<i>adr</i>	The address
<i>value</i>	The value

**Returns**

Zero if register found, non-zero if not found

**Warning**

This function is sub-optimal, if you use it heavily add an internal function to register.c

**See also**

[e\\_po3030k\\_get\\_register](#)

**7.27.3.32 int e\_po3030k\_set\_sampling\_mode ( int *mode* )**

Set the camera sampling mode

**Warning**

This is an internal function, use e\_po3030k\_config\_cam

**Parameters**

<i>mode</i>	The given sampling mode
-------------	-------------------------

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 28 and [e\\_po3030k\\_config\\_cam](#)

**7.27.3.33 void e\_po3030k\_set\_sepia ( int status )**

Enable/Disable Sepia color

**Parameters**

<i>status</i>	Set <i>status</i> to 1 to enable, 0 to disable
---------------	--

**See also**

Datasheet p.34 and 74

**7.27.3.34 void e\_po3030k\_set\_sepia\_tone ( unsigned char cb, unsigned char cr )**

Set The color tone at sepia color condition

**Parameters**

<i>cb</i>	Cb tone
<i>cr</i>	Cr tone

**See also**

[e\\_po3030k\\_set\\_sepia](#) and Datasheet p. 41 and 74

**7.27.3.35 int e\_po3030k\_set\_speed ( int mode )**

Set the camera speed

**Warning**

This is an internal function, use [e\\_po3030k\\_config\\_cam](#)



## Parameters

<i>mode</i>	The given speed
-------------	-----------------

## Returns

Zero if OK, non-zero if unknow mode

## See also

Datasheet p. 26 and [e\\_po3030k\\_config\\_cam](#)

7.27.3.36 int e\_po3030k\_set\_vsync ( unsigned int *start*, unsigned int *stop*, unsigned int *col* )

Set the camera window VSYNC coordinate

## Warning

This is an internal function, use e\_po3030k\_ConfigCam

## Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row
<i>col</i>	The start/stop column

## Returns

Zero if OK, non-zero if an error occur

## See also

Datasheet p.42-43, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

7.27.3.37 int e\_po3030k\_set\_weight\_win ( unsigned int *x1*, unsigned int *x2*, unsigned int *y1*, unsigned int *y2* )

Set the Weighting Window coordinate

## Parameters

<i>x1</i>	The X1 coordinate $\in \{211, x2\}$
<i>x2</i>	The X2 coordinate $\in \{x1 + 1, 423\}$
<i>y1</i>	The Y1 coordinate $\in \{160, y2\}$
<i>y2</i>	The Y2 coordinate $\in \{y1 + 1, 319\}$

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 49

**7.27.3.38 void e\_po3030k\_set\_ww ( unsigned char *ww* )**

Set the Center weight (Back Light compensation) Control parameter

**Parameters**

<i>ww</i>	Center weight ( 4 lower bits only )
-----------	-------------------------------------

**See also**

Datasheet p.44

**7.27.3.39 int e\_po3030k\_set\_wx ( unsigned int *start*, unsigned int *stop* )**

Set the camera window X coordinate

**Warning**

This is an internal function, use e\_po3030k\_ConfigCam

**Parameters**

<i>start</i>	The start column
<i>stop</i>	The stop column

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.20-21, [e\\_po3030k\\_write\\_cam\\_registers](#), [e\\_po3030k\\_set\\_wy](#) and [e\\_po3030k\\_config\\_cam](#)

**7.27.3.40 int e\_po3030k\_set\_wy ( unsigned int *start*, unsigned int *stop* )**

Set the camera window Y coordinate

**Warning**

This is an internal function, use e\_po3030k\_ConfigCam

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.20-21, [e\\_po3030k\\_WriteCamRegisters](#), [e\\_po3030k\\_SetWX](#) and [e\\_po3030k\\_ConfigCam](#)

7.27.3.41 `int e_po3030k_sync_register_array ( unsigned char start, unsigned char stop )`

Write every known register between address start and stop (inclusivly).

**Warning**

It's better to set the configuration with appropriate functions and then write all registers with [e\\_po3030k\\_WriteCamRegisters](#)

**Parameters**

<i>start</i>	The beginning address of the write
<i>stop</i>	The last write address

**Returns**

The number of register written

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.27.3.42 `void e_po3030k_write_cam_registers ( void )`

The Po3030k module keep in memory the state of each register the camera has. When you configure the camera, it only alter the internal register state, not the camera one. This function write the internal register state in the camera.

**See also**

[e\\_po3030k\\_read\\_cam\\_registers](#)

7.27.3.43 void e\_po3030k\_write\_gamma\_coef ( void )

This special function write directly the Gamma coefficient and Gamma color select into camera register.

#### Warning

This function need extra care from the user

#### See also

[e\\_po3030k\\_set\\_gamma\\_coef](#)

## 7.28 camera/fast\_2\_timer/e\_po3030k\_registers.c File Reference

Manage po3030k registers (two timers)

```
#include "../I2C/e_I2C_protocol.h"
#include "e_po3030k.h"
```

#### Macros

- #define MCLK ((long) 14745600) /\* 14.7456Mhz \*/
- #define MCLK\_P\_NS 0.067816840278 /\* Master clock period in ns \*/
- #define ARRAY\_ORIGINE\_X 210
- #define ARRAY\_ORIGINE\_Y 7
- #define BASE\_D1 0
- #define BASE\_D2 0
- #define BASE\_D3 0
- #define BASE\_D4 0
- #define DEVICE\_ID 0xDC
- #define FRAME\_WIDTH 0x0353
- #define FRAME\_HEIGHT 0x01e8
- #define WINDOW\_X1\_BASE 9
- #define WINDOW\_Y1\_BASE 13
- #define WINDOW\_X2\_BASE 17
- #define WINDOW\_Y2\_BASE 21
- #define BIAS\_BASE 25
- #define COLGAIN\_BASE 29
- #define INTEGR\_BASE 39
- #define SPEED\_ADDR (45 - BASE\_D1)
- #define MIRROR\_BASE 47
- #define SAMPLING\_ADDR ( 51 - BASE\_D2 )
- #define ADCOFF\_BASE ( 53 - BASE\_D2 )
- #define FLICKM\_BASE ( 55 - BASE\_D2 )
- #define FLICKP\_BASE 59
- #define COLOR\_M\_ADDR (67 - BASE\_D3)
- #define MODE\_R5G6B5 0x08
- #define MODE\_YUV 0x02
- #define MODE\_GRAYSCALE 0x0c
- #define SEPIA\_BASE 69

- #define [LENSG\\_BASE](#) 73
- #define [EDGE\\_BASE](#) 79
- #define [GAMMA\\_BASE](#) 83
- #define [COLOR\\_COEF\\_BASE](#) 107
- #define [CBCRGAIN\\_BASE](#) 125
- #define [BRICTR\\_BASE](#) 129
- #define [SEPIATONE\\_BASE](#) 133
- #define [VSYNCSTART\\_BASE](#) (145 - [BASE\\_D4](#))
- #define [VSYNCSTOP\\_BASE](#) (149 - [BASE\\_D4](#))
- #define [VSYNCCOL\\_BASE](#) (153 - [BASE\\_D4](#))
- #define [WW\\_BASE](#) 157
- #define [AWVAETOL\\_BASE](#) 159
- #define [AESPEED\\_BASE](#) 161
- #define [EXPOSURE\\_BASE](#) 163
- #define [REFREPO\\_BASE](#) 169
- #define [MINMAXEXP\\_BASE](#) 175
- #define [MINMAXAWB\\_BASE](#) 183
- #define [WEIGHWIN\\_BASE](#) 195
- #define [AWBAEENABLE\\_BASE](#) 211
- #define [GAMMASELCOL\\_BASE](#) 215
- #define [NB\\_REGISTERS](#) (sizeof([cam\\_reg](#))/(2\*sizeof([cam\\_reg](#)[0])))

## Functions

- void [e\\_po3030k\\_write\\_cam\\_registers](#) (void)
- void [e\\_po3030k\\_read\\_cam\\_registers](#) (void)
- int [e\\_po3030k\\_set\\_color\\_mode](#) (int mode)
- int [e\\_po3030k\\_set\\_sampling\\_mode](#) (int mode)
- int [e\\_po3030k\\_set\\_speed](#) (int mode)
- int [e\\_po3030k\\_set\\_wx](#) (unsigned int start, unsigned int stop)
- int [e\\_po3030k\\_set\\_wy](#) (unsigned int start, unsigned int stop)
- int [e\\_po3030k\\_set\\_vsync](#) (unsigned int start, unsigned int stop, unsigned int col)
- void [e\\_po3030k\\_set\\_mirror](#) (int vertical, int horizontal)
- int [e\\_po3030k\\_set\\_register](#) (unsigned char adr, unsigned char value)
- int [e\\_po3030k\\_get\\_register](#) (unsigned char adr, unsigned char \*value)
- void [e\\_po3030k\\_set\\_bias](#) (unsigned char pixbias, unsigned char opbias)
- int [e\\_po3030k\\_set\\_color\\_gain](#) (unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)
- void [e\\_po3030k\\_set\\_integr\\_time](#) (unsigned long time)
- void [e\\_po3030k\\_set\\_adc\\_offset](#) (unsigned char offset)
- void [e\\_po3030k\\_set\\_sepia](#) (int status)
- void [e\\_po3030k\\_set\\_lens\\_gain](#) (unsigned char red, unsigned char green, unsigned char blue)
- void [e\\_po3030k\\_set\\_edge\\_prop](#) (unsigned char gain, unsigned char tresh)
- void [e\\_po3030k\\_set\\_gamma\\_coef](#) (unsigned char array[12], char color)
- void [e\\_po3030k\\_write\\_gamma\\_coef](#) (void)
- int [e\\_po3030k\\_sync\\_register\\_array](#) (unsigned char start, unsigned char stop)
- void [e\\_po3030k\\_SetColorMatrix](#) (unsigned char array[3 \* 3])
- void [e\\_po3030k\\_set\\_cb\\_cr\\_gain](#) (unsigned char cg11c, unsigned char cg22c)
- void [e\\_po3030k\\_set\\_brigh\\_contr](#) (unsigned char bright, unsigned char contrast)
- void [e\\_po3030k\\_set\\_sepia\\_tone](#) (unsigned char cb, unsigned char cr)
- void [e\\_po3030k\\_set\\_ww](#) (unsigned char ww)
- void [e\\_po3030k\\_set\\_awb\\_ae\\_tol](#) (unsigned char awbm, unsigned char aem)
- void [e\\_po3030k\\_set\\_ae\\_speed](#) (unsigned char b, unsigned char d)
- void [e\\_po3030k\\_set\\_exposure](#) (long t)

- void `e_po3030k_set_ref_exposure` (unsigned char exp)
- void `e_po3030k_set_max_min_exp` (unsigned int max, unsigned int min)
- void `e_po3030k_set_max_min_awb` (unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob)
- int `e_po3030k_set_weight_win` (unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2)
- void `e_po3030k_set_awb_ae` (int awb, int ae)
- void `e_po3030k_set_flicker_mode` (int manual)
- void `e_po3030k_set_flicker_detection` (int hz50, int hz60)
- static long `po3030k_get_pixelclock` (void)
- int `e_po3030k_set_flicker_man_set` (int hz50, int hz60, int fdm, int fk, int tol)

## Variables

- static unsigned char `cam_reg` []

### 7.28.1 Detailed Description

Manage po3030k registers (two timers)

#### Author

Philippe Rétornaz

### 7.28.2 Macro Definition Documentation

7.28.2.1 `#define ADCOFF_BASE ( 53 - BASE_D2 )`

7.28.2.2 `#define AESPEED_BASE 161`

7.28.2.3 `#define ARRAY_ORIGINE_X 210`

7.28.2.4 `#define ARRAY_ORIGINE_Y 7`

7.28.2.5 `#define AWBAENABLE_BASE 211`

7.28.2.6 `#define AWVAETOL_BASE 159`

7.28.2.7 `#define BASE_D1 0`

7.28.2.8 `#define BASE_D2 0`

7.28.2.9 `#define BASE_D3 0`

7.28.2.10 `#define BASE_D4 0`

7.28.2.11 `#define BIAS_BASE 25`

7.28.2.12 #define BRICTR\_BASE 129

7.28.2.13 #define CBCRGAIN\_BASE 125

7.28.2.14 #define COLGAIN\_BASE 29

7.28.2.15 #define COLOR\_COEF\_BASE 107

7.28.2.16 #define COLOR\_M\_ADDR (67 - BASE\_D3)

7.28.2.17 #define DEVICE\_ID 0xDC

7.28.2.18 #define EDGE\_BASE 79

7.28.2.19 #define EXPOSURE\_BASE 163

7.28.2.20 #define FLICKM\_BASE ( 55 - BASE\_D2 )

7.28.2.21 #define FLICKP\_BASE 59

7.28.2.22 #define FRAME\_HEIGHT 0x01e8

7.28.2.23 #define FRAME\_WIDTH 0x0353

7.28.2.24 #define GAMMA\_BASE 83

7.28.2.25 #define GAMMASELCOL\_BASE 215

7.28.2.26 #define INTEGR\_BASE 39

7.28.2.27 #define LENS\_G\_BASE 73

7.28.2.28 #define MCLK ((long) 14745600) /\* 14.7456Mhz \*/

7.28.2.29 #define MCLK\_P\_NS 0.067816840278 /\* Master clock period in ns \*/

7.28.2.30 #define MINMAXAWB\_BASE 183

7.28.2.31 #define MINMAXEXP\_BASE 175

7.28.2.32 #define MIRROR\_BASE 47

7.28.2.33 #define MODE\_GRAYSCALE 0x0c

7.28.2.34 #define MODE\_R5G6B5 0x08

7.28.2.35 `#define MODE_YUV 0x02`

7.28.2.36 `#define NB_REGISTERS (sizeof(cam_reg)/(2*sizeof(cam_reg[0])))`

7.28.2.37 `#define REFREPO_BASE 169`

7.28.2.38 `#define SAMPLING_ADDR ( 51 - BASE_D2 )`

7.28.2.39 `#define SEPIA_BASE 69`

7.28.2.40 `#define SEPIATONE_BASE 133`

7.28.2.41 `#define SPEED_ADDR (45 - BASE_D1)`

7.28.2.42 `#define VSYNCCOL_BASE (153 - BASE_D4)`

7.28.2.43 `#define VSYNCSTART_BASE (145 - BASE_D4)`

7.28.2.44 `#define VSYNCSTOP_BASE (149 - BASE_D4)`

7.28.2.45 `#define WEIGHWIN_BASE 195`

7.28.2.46 `#define WINDOW_X1_BASE 9`

7.28.2.47 `#define WINDOW_X2_BASE 17`

7.28.2.48 `#define WINDOW_Y1_BASE 13`

7.28.2.49 `#define WINDOW_Y2_BASE 21`

7.28.2.50 `#define WW_BASE 157`

### 7.28.3 Function Documentation

7.28.3.1 `int e_po3030k_get_register ( unsigned char adr, unsigned char * value )`

Get the register *adr* value

#### Parameters

<i>adr</i>	The address
<i>value</i>	The pointer to the value to write to



**Returns**

Zero if register found, non-zero if not found

**Warning**

This function is sub-optimal, if you use it heavily add an internal function to register.c

**See also**

[e\\_po3030k\\_set\\_register](#)

**7.28.3.2 void e\_po3030k\_read\_cam\_registers ( void )**

Read the camera register

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

**7.28.3.3 void e\_po3030k\_set\_adc\_offset ( unsigned char *offset* )**

Set the Analog to Digital Converter offset

**Parameters**

<i>offset</i>	The offset
---------------	------------

**See also**

Datasheet p.28

**7.28.3.4 void e\_po3030k\_set\_ae\_speed ( unsigned char *b*, unsigned char *d* )**

Set AE speed

**Parameters**

<i>b</i>	AE speed factor when exposure time is decreasing ( 4 lower bits only )
<i>d</i>	AE speed factor when exposure time is increasing ( 4 lower bits only )

**See also**

Datasheet p.44

7.28.3.5 void e\_po3030k\_set\_awb\_ae ( int *awb*, int *ae* )

Enable/Disable AWB and AE

Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

See also

Datasheet p. 50

7.28.3.6 void e\_po3030k\_set\_awb\_ae\_tol ( unsigned char *awbm*, unsigned char *aem* )

Set AWB/AE tolerance margin

Parameters

<i>awbm</i>	AWV Margin ( 4 lower bits only )
<i>aem</i>	AW Margin ( 4 lower bits only )

See also

Datasheet p.44

7.28.3.7 void e\_po3030k\_set\_bias ( unsigned char *pixbias*, unsigned char *opbias* )

Set the Pixel and amplifcator bias Increasing the bias produce better image quality, but increase camera's power consumption

Parameters

<i>pixbias</i>	The pixel bias
<i>opbias</i>	The Amplificator bias

See also

Datasheet p.22

7.28.3.8 void e\_po3030k\_set\_brigh\_contr ( unsigned char *bright*, unsigned char *contrast* )

Set the Brightness & Contrast

## Parameters

<i>bright</i>	The Brightness ( signed 1+7bits fixed point format )
<i>contrast</i>	The Contrast

## See also

Datasheet p. 40

7.28.3.9 void e\_po3030k\_set\_cb\_cr\_gain ( unsigned char *cg11c*, unsigned char *cg22c* )

Set The color gain ( Cb/Cr )

## Parameters

<i>cg11c</i>	Cb gain ( Sign[7]   Integer[6:5]   fractional[4:0] )
<i>cg22c</i>	Cr gain ( Sign[7]   Integer[6:5]   fractional[4:0] )

## See also

Datasheet p. 40

7.28.3.10 int e\_po3030k\_set\_color\_gain ( unsigned char *global*, unsigned char *red*, unsigned char *green1*, unsigned char *green2*, unsigned char *blue* )

Set the gains of the camera

## Parameters

<i>global</i>	The global gain $\in \{0, 79\}$
<i>red</i>	The red pixel's gain (fixed point [2:6] format)
<i>green1</i>	The green pixel near read one gain ([2:6] format)
<i>green2</i>	The green pixel near blue one gain ([2:6] format)
<i>blue</i>	The blue pixel's gain ([2:6] format)

## Returns

Zero if OK, non-zero if an error occur

## See also

Datasheet p.23-24

### 7.28.3.11 int e\_po3030k\_set\_color\_mode ( int *mode* )

Set the camera color mode

#### Warning

This is an internal function, use `e_po3030k_config_cam`

#### Parameters

<i>mode</i>	The color mode
-------------	----------------

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p. 31, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

### 7.28.3.12 void e\_po3030k\_set\_edge\_prop ( unsigned char *gain*, unsigned char *tresh* )

Set Edge properties

#### Parameters

<i>gain</i>	Edge gain & moire factor (fixed point [2:3] format)
<i>tresh</i>	Edge Enhancement threshold

#### See also

Datasheet p.36

### 7.28.3.13 void e\_po3030k\_set\_exposure ( long *t* )

Set exposure time

#### Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

#### Warning

Only writable if AE is disabled

**See also**

Datasheet p.45

7.28.3.14 void e\_po3030k\_set\_flicker\_detection ( int *hz50*, int *hz60* )

Set the 50/60Hz flicker detection

**Parameters**

<i>hz50</i>	Non-zero mean 50Hz flicker detection enabled (default disabled)
<i>hz60</i>	Non-zero mean 60Hz flicker detection enabled (default disabled)

**Warning**

If Automatic mode is enabled and both 50Hz and 60Hz are disabled, camera will enable both. By default, the camera automatically detect 50 and 60Hz flicker.

**See also**

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

7.28.3.15 int e\_po3030k\_set\_flicker\_man\_set ( int *hz50*, int *hz60*, int *fdm*, int *fk*, int *tol* )

Set the camera's manual flicker's detection setting

**Parameters**

<i>hz50</i>	The Hz for the 50Hz detection
<i>hz60</i>	The Hz for the 60Hz detection
<i>fdm</i>	Flicker duration mode
<i>fk</i>	Flicker count step
<i>tol</i>	Flicker tolerance

**Warning**

You must have set the mode ( image size, color ) before calling this function

**Returns**

Non-zero if an error occur, 0 if OK

**See also**

Datasheet p.29-30 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#)

7.28.3.16 void e\_po3030k\_set\_flicker\_mode ( int *manual* )

Set flicker detection mode

## Parameters

<i>manual</i>	Non-zero mean manual mode is enabled ( default automatic mode enabled )
---------------	---

## See also

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

7.28.3.17 void `e_po3030k_set_gamma_coef ( unsigned char array[12], char color )`

Set gamma coefficient

## Warning

This feature need extra care from the user

## Parameters

<i>array</i>	Gamma coefficient array
<i>color</i>	First two bytes : - 0b01 => Green <ul style="list-style-type: none"> <li>• 0b00 =&gt; Red</li> <li>• else =&gt; Blue</li> </ul>

## See also

Datasheet p. 38, 50, 57-58 and `e_po3030k_WriteGammaCoef`

7.28.3.18 void `e_po3030k_set_integr_time ( unsigned long time )`

Set the pixel intergration time This is counted in line-time interval. See dataset p.25 for more information

## Parameters

<i>time</i>	The integration time ( fixed point [14:6] format )
-------------	--

## See also

Datasheet p.25

7.28.3.19 void `e_po3030k_set_lens_gain ( unsigned char red, unsigned char green, unsigned char blue )`

Set lens shading gain

## Parameters

<i>red</i>	Lens gain for red pixel $\in \{0, 15\}$
<i>green</i>	Lens gain for green pixel $\in \{0, 15\}$
<i>blue</i>	Lens gain for blue pixel $\in \{0, 15\}$

## See also

Datasheet p.36

7.28.3.20 void e\_po3030k\_set\_max\_min\_awb ( unsigned char *minb*, unsigned char *maxb*, unsigned char *minr*, unsigned char *maxr*, unsigned char *ratior*, unsigned char *ratiob* )

Set the minimum and maximum red and blue gain in AWB mode

## Parameters

<i>minb</i>	The minimum blue gain
<i>maxb</i>	The maximum blue gain
<i>minr</i>	The minimum red gain
<i>maxr</i>	The maximum red gain
<i>ratior</i>	The red gain ratio
<i>ratiob</i>	The blue gain ratio

## See also

Datasheet p. 47-48

7.28.3.21 void e\_po3030k\_set\_max\_min\_exp ( unsigned int *max*, unsigned int *min* )

Set the minimum and maximum exposure time in AE mode

## Parameters

<i>min</i>	The minimum exposure time
<i>max</i>	The maximum exposure time

## See also

Datasheet p.46-47

7.28.3.22 void e\_po3030k\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

**Parameters**

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

**See also**

Datasheet p.27

**7.28.3.23 void e\_po3030k\_set\_ref\_exposure ( unsigned char *exp* )**

Set the reference exposure. The average brightness which the AE should have

**Parameters**

<i>exp</i>	The target exposure level
------------	---------------------------

**See also**

Datasheet p.45

**7.28.3.24 int e\_po3030k\_set\_register ( unsigned char *adr*, unsigned char *value* )**

Set the register *adr* to value *value*

**Parameters**

<i>adr</i>	The address
<i>value</i>	The value

**Returns**

Zero if register found, non-zero if not found

**Warning**

This function is sub-optimal, if you use it heavily add an internal function to register.c

**See also**

[e\\_po3030k\\_get\\_register](#)

**7.28.3.25 int e\_po3030k\_set\_sampling\_mode ( int *mode* )**

Set the camera sampling mode

**Warning**

This is an internal function, use `e_po3030k_config_cam`



**Parameters**

<i>mode</i>	The given sampling mode
-------------	-------------------------

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 28 and [e\\_po3030k\\_config\\_cam](#)

**7.28.3.26 void e\_po3030k\_set\_sepia ( int status )**

Enable/Disable Sepia color

**Parameters**

<i>status</i>	Set <i>status</i> to 1 to enable, 0 to disable
---------------	--

**See also**

Datasheet p.34 and 74

**7.28.3.27 void e\_po3030k\_set\_sepia\_tone ( unsigned char cb, unsigned char cr )**

Set The color tone at sepia color condition

**Parameters**

<i>cb</i>	Cb tone
<i>cr</i>	Cr tone

**See also**

[e\\_po3030k\\_set\\_sepia](#) and Datasheet p. 41 and 74

**7.28.3.28 int e\_po3030k\_set\_speed ( int mode )**

Set the camera speed

**Warning**

This is an internal function, use [e\\_po3030k\\_config\\_cam](#)

**Parameters**

<i>mode</i>	The given speed
-------------	-----------------

**Returns**

Zero if OK, non-zero if unknow mode

**See also**

Datasheet p. 26 and [e\\_po3030k\\_config\\_cam](#)

### 7.28.3.29 int e\_po3030k\_set\_vsync ( unsigned int *start*, unsigned int *stop*, unsigned int *col* )

Set the camera window VSYNC coordinate

**Warning**

This is an internal function, use e\_po3030k\_ConfigCam

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row
<i>col</i>	The start/stop column

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.42-43, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

### 7.28.3.30 int e\_po3030k\_set\_weight\_win ( unsigned int *x1*, unsigned int *x2*, unsigned int *y1*, unsigned int *y2* )

Set the Weighting Window coordinate

**Parameters**

<i>x1</i>	The X1 coordinate $\in \{211, x2\}$
<i>x2</i>	The X2 coordinate $\in \{x1 + 1, 423\}$
<i>y1</i>	The Y1 coordinate $\in \{160, y2\}$
<i>y2</i>	The Y2 coordinate $\in \{y1 + 1, 319\}$

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 49

**7.28.3.31 void e\_po3030k\_set\_ww ( unsigned char *ww* )**

Set the Center weight (Back Light compensation) Control parameter

**Parameters**

<i>ww</i>	Center weight ( 4 lower bits only )
-----------	-------------------------------------

**See also**

Datasheet p.44

**7.28.3.32 int e\_po3030k\_set\_wx ( unsigned int *start*, unsigned int *stop* )**

Set the camera window X coordinate

**Warning**

This is an internal function, use e\_po3030k\_ConfigCam

**Parameters**

<i>start</i>	The start column
<i>stop</i>	The stop column

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.20-21, [e\\_po3030k\\_write\\_cam\\_registers](#), [e\\_po3030k\\_set\\_wy](#) and [e\\_po3030k\\_config\\_cam](#)

**7.28.3.33 int e\_po3030k\_set\_wy ( unsigned int *start*, unsigned int *stop* )**

Set the camera window Y coordinate

**Warning**

This is an internal function, use e\_po3030k\_ConfigCam

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.20-21, `e_po3030k_WriteCamRegisters`, `e_po3030k_SetWX` and `e_po3030k_ConfigCam`

### 7.28.3.34 void `e_po3030k_SetColorMatrix` ( unsigned char *array*[3 \*3] )

Set color correction coefficient

**Parameters**

<i>array</i>	Color coefficient matrix (3x3), sign[7]   integer [6:5]   fractional [4:0]
--------------	--

**See also**

Datasheet p. 39

### 7.28.3.35 int `e_po3030k_sync_register_array` ( unsigned char *start*, unsigned char *stop* )

Write every known register between address *start* and *stop* (inclusively).

**Warning**

It's better to set the configuration with appropriate functions and then write all registers with `e_po3030k_WriteCamRegisters`

**Parameters**

<i>start</i>	The beginning address of the write
<i>stop</i>	The last write address

**Returns**

The number of register written

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.28.3.36 void e\_po3030k\_write\_cam\_registers ( void )

The Po3030k module keep in memory the state of each register the camera has. When you configure the camera, it only alter the internal register state, not the camera one. This function write the internal register state in the camera.

See also

[e\\_po3030k\\_read\\_cam\\_registers](#)

7.28.3.37 void e\_po3030k\_write\_gamma\_coef ( void )

This special function write directly the Gamma coefficient and Gamma color select into camera register.

Warning

This function need extra care from the user

See also

[e\\_po3030k\\_set\\_gamma\\_coef](#)

7.28.3.38 static long po3030k\_get\_pixelclock ( void ) [static]

## 7.28.4 Variable Documentation

7.28.4.1 unsigned char cam\_reg[] [static]

## 7.29 camera/fast\_2\_timer/e\_po6030k.h File Reference

PO6030k library header.

```
#include "e_poxxxx.h"
```

### Macros

- #define [PO\\_6030\\_MODE\\_VGA](#) 0x20
- #define [PO\\_6030\\_MODE\\_QVGA](#) 0x40
- #define [PO\\_6030\\_MODE\\_QQVGA](#) 0x80
- #define [BAYER\\_CLOCK\\_1](#) 0x10
- #define [BAYER\\_CLOCK\\_2](#) 0x50
- #define [BAYER\\_CLOCK\\_4](#) 0x90
- #define [BAYER\\_CLOCK\\_8](#) 0xB0
- #define [BANK\\_A](#) 0x0
- #define [BANK\\_B](#) 0x1
- #define [BANK\\_C](#) 0x2
- #define [BANK\\_D](#) 0x3
- #define [PO\\_6030\\_SPEED\\_1](#) [BAYER\\_CLOCK\\_1](#)
- #define [PO\\_6030\\_SPEED\\_2](#) [BAYER\\_CLOCK\\_2](#)
- #define [PO\\_6030\\_SPEED\\_4](#) [BAYER\\_CLOCK\\_4](#)
- #define [PO\\_6030\\_SPEED\\_8](#) [BAYER\\_CLOCK\\_8](#)
- #define [e\\_po6030k\\_set\\_speed](#)(div) [e\\_po6030k\\_set\\_bayer\\_clkdiv](#)(div)
- #define [E\\_PO6030K\\_SKETCH\\_BW](#) 0
- #define [E\\_PO6030K\\_SKETCH\\_COLOR](#) 1

## Functions

- int [e\\_po6030k\\_config\\_cam](#) (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- int [e\\_po6030k\\_get\\_bytes\\_per\\_pixel](#) (int color\_mode)
- void [e\\_po6030k\\_set\\_color\\_mode](#) (unsigned char format)
- void [e\\_po6030k\\_set\\_bank](#) (unsigned char bank)
- void [e\\_po6030k\\_write\\_register](#) (unsigned char bank, unsigned char reg, unsigned char value)
- void [e\\_po6030k\\_set\\_bayer\\_clkdiv](#) (unsigned char div)
- void [e\\_po6030k\\_set\\_pclkdiv](#) (unsigned char div)
- int [e\\_po6030k\\_set\\_mode](#) (unsigned char format, unsigned char sampl\_mode)
- int [e\\_po6030k\\_set\\_wx](#) (unsigned int start, unsigned int stop)
- int [e\\_po6030k\\_set\\_wy](#) (unsigned int start, unsigned int stop)
- int [e\\_po6030k\\_set\\_vsync](#) (unsigned int start, unsigned int stop)
- void [e\\_po6030k\\_set\\_sketch\\_mode](#) (int mode)
- void [e\\_po6030k\\_set\\_mirror](#) (int vertical, int horizontal)
- void [e\\_po6030k\\_set\\_awb\\_ae](#) (int awb, int ae)
- void [e\\_po6030k\\_set\\_rgb\\_gain](#) (unsigned char r, unsigned char g, unsigned char b)
- void [e\\_po6030k\\_set\\_exposure](#) (unsigned long exp)

### 7.29.1 Detailed Description

PO6030k library header.

#### Author

Philippe Rétornaz

### 7.29.2 Macro Definition Documentation

7.29.2.1 `#define BANK_A 0x0`

7.29.2.2 `#define BANK_B 0x1`

7.29.2.3 `#define BANK_C 0x2`

7.29.2.4 `#define BANK_D 0x3`

7.29.2.5 `#define BAYER_CLOCK_1 0x10`

7.29.2.6 `#define BAYER_CLOCK_2 0x50`

7.29.2.7 `#define BAYER_CLOCK_4 0x90`

7.29.2.8 `#define BAYER_CLOCK_8 0xB0`

7.29.2.9 `#define e_po6030k_set_speed( div ) e_po6030k_set_bayer_clkdiv(div)`

- 7.29.2.10 #define E\_PO6030K\_SKETCH\_BW 0
- 7.29.2.11 #define E\_PO6030K\_SKETCH\_COLOR 1
- 7.29.2.12 #define PO\_6030\_MODE\_QQVGA 0x80
- 7.29.2.13 #define PO\_6030\_MODE\_QVGA 0x40
- 7.29.2.14 #define PO\_6030\_MODE\_VGA 0x20
- 7.29.2.15 #define PO\_6030\_SPEED\_1 BAYER\_CLOCK\_1
- 7.29.2.16 #define PO\_6030\_SPEED\_2 BAYER\_CLOCK\_2
- 7.29.2.17 #define PO\_6030\_SPEED\_4 BAYER\_CLOCK\_4
- 7.29.2.18 #define PO\_6030\_SPEED\_8 BAYER\_CLOCK\_8

### 7.29.3 Function Documentation

- 7.29.3.1 int e\_po6030k\_config\_cam ( unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode* )

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

#### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

#### Parameters

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

#### Returns

Zero if the settings are correct, non-zero if an error occur

7.29.3.2 `int e_p06030k_get_bytes_per_pixel ( int color_mode )`

Return the number of bytes per pixel in the given color mode



## Parameters

<i>color_mode</i>	The given color mode
-------------------	----------------------

## Returns

The number of bytes per pixel in the given color mode

7.29.3.3 void e\_po6030k\_set\_awb\_ae ( int *awb*, int *ae* )

Enable/Disable AWB and AE

## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

7.29.3.4 void e\_po6030k\_set\_bank ( unsigned char *bank* )

Set the camera register bank to use

## Parameters

<i>bank</i>	The bank used.
-------------	----------------

## See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.29.3.5 void e\_po6030k\_set\_bayer\_clkdiv ( unsigned char *div* )

Set the bayer clock divider

## Parameters

<i>div</i>	the register value
------------	--------------------

## See also

[BAYER\\_CLOCK\\_1](#), [BAYER\\_CLOCK\\_2](#), [BAYER\\_CLOCK\\_4](#), [BAYER\\_CLOCK\\_8](#)

7.29.3.6 void e\_po6030k\_set\_color\_mode ( unsigned char *format* )

7.29.3.7 void e\_po6030k\_set\_exposure ( unsigned long *exp* )

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

7.29.3.8 void e\_po6030k\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

## Parameters

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

7.29.3.9 int e\_po6030k\_set\_mode ( unsigned char *format*, unsigned char *sampl\_mode* )

Set camera sampling and color mode

## Parameters

<i>format</i>	The color format
<i>sampl_mode</i>	The sampling mode

## See also

[GREY\\_SCALE\\_MODE](#), [RGB\\_565\\_MODE](#), [YUV\\_MODE](#), [MODE\\_VGA](#), [MODE\\_QVGA](#), [MODE\\_QQVGA](#)

7.29.3.10 void e\_po6030k\_set\_pclkdiv ( unsigned char *div* )

Set the pixel clock divider

## Parameters

<i>div</i>	the register value
------------	--------------------

## Warning

The register value is dependant of the camera color and sample mode

7.29.3.11 void e\_po6030k\_set\_rgb\_gain ( unsigned char *r*, unsigned char *g*, unsigned char *b* )7.29.3.12 void e\_po6030k\_set\_sketch\_mode ( int *mode* )

Set the camera sketch mode (aka, sobel-filter)

**Parameters**

<i>mode</i>	The sketch mode
-------------	-----------------

**See also**

[E\\_PO6030K\\_SKETCH\\_BW](#), [E\\_PO6030K\\_SKETCH\\_COLOR](#)

**7.29.3.13 int e\_po6030k\_set\_vsync ( unsigned int *start*, unsigned int *stop* )**

Set the camera window VSYNC coordinate

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row

**Returns**

Zero if OK, non-zero if an error occur

**7.29.3.14 int e\_po6030k\_set\_wx ( unsigned int *start*, unsigned int *stop* )**

Set the camera window X coordinate

**Parameters**

<i>start</i>	The start column
<i>stop</i>	The stop column

**Returns**

Zero if OK, non-zero if an error occur

**7.29.3.15 int e\_po6030k\_set\_wy ( unsigned int *start*, unsigned int *stop* )**

Set the camera window Y coordinate

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row

**Returns**

Zero if OK, non-zero if an error occur

7.29.3.16 void e\_po6030k\_write\_register ( unsigned char *bank*, unsigned char *reg*, unsigned char *value* )

Set the register reg to value

**Parameters**

<i>bank</i>	The register bank
<i>reg</i>	The register address
<i>value</i>	The value to write

**See also**

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

**7.30 camera/fast\_2\_timer/e\_po6030k\_registers.c File Reference**

Manage po6030k registers (two timers)

```
#include "../I2C/e_I2C_protocol.h"
#include "e_po6030k.h"
```

**Macros**

- #define [MCLK](#) ((long) 14745600) /\* 14.7456Mhz \*/
- #define [MCLK\\_P\\_NS](#) 0.067816840278 /\* Master clock period in ns \*/
- #define [ARRAY\\_ORIGINE\\_X](#) 80
- #define [ARRAY\\_ORIGINE\\_Y](#) 8
- #define [DEVICE\\_ID](#) 0xDC
- #define [BANK\\_REGISTER](#) 0x3

**Functions**

- void [e\\_po6030k\\_set\\_bank](#) (unsigned char bank)
- void [e\\_po6030k\\_write\\_register](#) (unsigned char bank, unsigned char reg, unsigned char value)
- unsigned char [e\\_po6030k\\_read\\_register](#) (unsigned char bank, unsigned char reg)
- void [e\\_po6030k\\_set\\_bayer\\_clkdiv](#) (unsigned char div)
- void [e\\_po6030k\\_set\\_pclkdiv](#) (unsigned char div)
- static int [e\\_po6030k\\_set\\_sampl\\_gray](#) (unsigned char sample)
- static int [e\\_po6030k\\_set\\_sampl\\_color](#) (unsigned char sample)
- int [e\\_po6030k\\_set\\_mode](#) (unsigned char format, unsigned char sampl\_mode)
- int [e\\_po6030k\\_set\\_wx](#) (unsigned int start, unsigned int stop)
- int [e\\_po6030k\\_set\\_wy](#) (unsigned int start, unsigned int stop)
- int [e\\_po6030k\\_set\\_vsync](#) (unsigned int start, unsigned int stop)
- void [e\\_po6030k\\_set\\_sketch\\_mode](#) (int mode)
- void [e\\_po6030k\\_set\\_mirror](#) (int vertical, int horizontal)
- void [e\\_po6030k\\_set\\_awb\\_ae](#) (int awb, int ae)
- void [e\\_po6030k\\_set\\_rgb\\_gain](#) (unsigned char r, unsigned char g, unsigned char b)
- void [e\\_po6030k\\_set\\_exposure](#) (unsigned long exp)

### 7.30.1 Detailed Description

Manage po6030k registers (two timers)

Author

Philippe Rétornaz

### 7.30.2 Macro Definition Documentation

7.30.2.1 `#define ARRAY_ORIGINE_X 80`

7.30.2.2 `#define ARRAY_ORIGINE_Y 8`

7.30.2.3 `#define BANK_REGISTER 0x3`

7.30.2.4 `#define DEVICE_ID 0xDC`

7.30.2.5 `#define MCLK ((long) 14745600) /* 14.7456Mhz */`

7.30.2.6 `#define MCLK_P_NS 0.067816840278 /* Master clock period in ns */`

### 7.30.3 Function Documentation

7.30.3.1 `unsigned char e_po6030k_read_register ( unsigned char bank, unsigned char reg )`

Read the register *reg*.

Parameters

<i>bank</i>	The register bank
<i>reg</i>	The register address

Returns

The register value

See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.30.3.2 `void e_po6030k_set_awb_ae ( int awb, int ae )`

Enable/Disable AWB and AE

## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

7.30.3.3 void e\_po6030k\_set\_bank ( unsigned char *bank* )

Set the camera register bank to use

## Parameters

<i>bank</i>	The bank used.
-------------	----------------

## See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.30.3.4 void e\_po6030k\_set\_bayer\_clkdiv ( unsigned char *div* )

Set the bayer clock divider

## Parameters

<i>div</i>	the register value
------------	--------------------

## See also

[BAYER\\_CLOCK\\_1](#), [BAYER\\_CLOCK\\_2](#), [BAYER\\_CLOCK\\_4](#), [BAYER\\_CLOCK\\_8](#)

7.30.3.5 void e\_po6030k\_set\_exposure ( unsigned long *exp* )

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

7.30.3.6 void e\_po6030k\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

## Parameters

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

7.30.3.7 `int e_po6030k_set_mode ( unsigned char format, unsigned char sampl_mode )`

Set camera sampling and color mode

Parameters

<i>format</i>	The color format
<i>sampl_mode</i>	The sampling mode

See also

[GREY\\_SCALE\\_MODE](#), [RGB\\_565\\_MODE](#), [YUV\\_MODE](#), [MODE\\_VGA](#), [MODE\\_QVGA](#), [MODE\\_QQVGA](#)

7.30.3.8 `void e_po6030k_set_pclkdiv ( unsigned char div )`

Set the pixel clock divider

Parameters

<i>div</i>	the register value
------------	--------------------

Warning

The register value is dependant of the camera color and sample mode

7.30.3.9 `void e_po6030k_set_rgb_gain ( unsigned char r, unsigned char g, unsigned char b )`

7.30.3.10 `static int e_po6030k_set_sampl_color ( unsigned char sample ) [static]`

7.30.3.11 `static int e_po6030k_set_sampl_gray ( unsigned char sample ) [static]`

7.30.3.12 `void e_po6030k_set_sketch_mode ( int mode )`

Set the camera sketch mode (aka, sobel-filter)

Parameters

<i>mode</i>	The sketch mode
-------------	-----------------

See also

[E\\_PO6030K\\_SKETCH\\_BW](#), [E\\_PO6030K\\_SKETCH\\_COLOR](#)

7.30.3.13 `int e_po6030k_set_vsync ( unsigned int start, unsigned int stop )`

Set the camera window VSYNC coordinate



## Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row

## Returns

Zero if OK, non-zero if an error occur

7.30.3.14 int e\_po6030k\_set\_wx ( unsigned int *start*, unsigned int *stop* )

Set the camera window X coordinate

## Parameters

<i>start</i>	The start column
<i>stop</i>	The stop column

## Returns

Zero if OK, non-zero if an error occur

7.30.3.15 int e\_po6030k\_set\_wy ( unsigned int *start*, unsigned int *stop* )

Set the camera window Y coordinate

## Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row

## Returns

Zero if OK, non-zero if an error occur

7.30.3.16 void e\_po6030k\_write\_register ( unsigned char *bank*, unsigned char *reg*, unsigned char *value* )

Set the register reg to value

## Parameters

<i>bank</i>	The register bank
<i>reg</i>	The register address
<i>value</i>	The value to write

See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

## 7.31 camera/fast\_2\_timer/e\_po8030d.h File Reference

P08030d library header.

```
#include "e_poxxxx.h"
```

### Macros

- `#define PO_8030_MODE_VGA 0x20`
- `#define PO_8030_MODE_QVGA 0x40`
- `#define PO_8030_MODE_QQVGA 0x80`
- `#define PO_8030_BAYER_CLOCK_1 0x00`
- `#define PO_8030_BAYER_CLOCK_2 0x02`
- `#define PO_8030_BAYER_CLOCK_4 0x04`
- `#define PO_8030_BAYER_CLOCK_8 0x05`
- `#define BANK_A 0x0`
- `#define BANK_B 0x1`
- `#define BANK_C 0x2`
- `#define BANK_D 0x3`
- `#define PO_8030_SPEED_1 PO_8030_BAYER_CLOCK_1`
- `#define PO_8030_SPEED_2 PO_8030_BAYER_CLOCK_2`
- `#define PO_8030_SPEED_4 PO_8030_BAYER_CLOCK_4`
- `#define PO_8030_SPEED_8 PO_8030_BAYER_CLOCK_8`
- `#define e_po8030d_set_speed(div) e_po8030d_set_bayer_clkdiv(div)`
- `#define E_PO8030D_SKETCH_BW 0`
- `#define E_PO8030D_SKETCH_COLOR 1`

### Functions

- `int e_po8030d_config_cam` (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- `int e_po8030d_get_bytes_per_pixel` (int color\_mode)
- `void e_po8030d_set_color_mode` (unsigned char format)
- `void e_po8030d_set_bank` (unsigned char bank)
- `void e_po8030d_write_register` (unsigned char bank, unsigned char reg, unsigned char value)
- `void e_po8030d_set_bayer_clkdiv` (unsigned char div)
- `void e_po8030d_set_pclkdiv` (unsigned char div)
- `int e_po8030d_set_mode` (unsigned char format, unsigned char sampl\_mode)
- `int e_po8030d_set_wx` (unsigned int start, unsigned int stop)
- `int e_po8030d_set_wy` (unsigned int start, unsigned int stop)
- `int e_po8030d_set_vsync` (unsigned int start, unsigned int stop)
- `void e_po8030d_set_sketch_mode` (int mode)
- `void e_po8030d_set_mirror` (int vertical, int horizontal)
- `void e_po8030d_set_awb_ae` (int awb, int ae)
- `void e_po8030d_set_rgb_gain` (unsigned char r, unsigned char g, unsigned char b)
- `void e_po8030d_set_exposure` (unsigned long exp)
- `int testWriteReg` (unsigned char bank)
- `int init_po8030` ()
- `void e_po8030d_set_brightness` (signed char value)

### 7.31.1 Detailed Description

P08030d library header.

Author

Stefano Morgani

### 7.31.2 Macro Definition Documentation

7.31.2.1 #define BANK\_A 0x0

7.31.2.2 #define BANK\_B 0x1

7.31.2.3 #define BANK\_C 0x2

7.31.2.4 #define BANK\_D 0x3

7.31.2.5 #define e\_po8030d\_set\_speed( div ) e\_po8030d\_set\_bayer\_clkdiv(div)

7.31.2.6 #define E\_PO8030D\_SKETCH\_BW 0

7.31.2.7 #define E\_PO8030D\_SKETCH\_COLOR 1

7.31.2.8 #define PO\_8030\_BAYER\_CLOCK\_1 0x00

7.31.2.9 #define PO\_8030\_BAYER\_CLOCK\_2 0x02

7.31.2.10 #define PO\_8030\_BAYER\_CLOCK\_4 0x04

7.31.2.11 #define PO\_8030\_BAYER\_CLOCK\_8 0x05

7.31.2.12 #define PO\_8030\_MODE\_QQVGA 0x80

7.31.2.13 #define PO\_8030\_MODE\_QVGA 0x40

7.31.2.14 #define PO\_8030\_MODE\_VGA 0x20

7.31.2.15 #define PO\_8030\_SPEED\_1 PO\_8030\_BAYER\_CLOCK\_1

7.31.2.16 #define PO\_8030\_SPEED\_2 PO\_8030\_BAYER\_CLOCK\_2

7.31.2.17 #define PO\_8030\_SPEED\_4 PO\_8030\_BAYER\_CLOCK\_4

7.31.2.18 #define PO\_8030\_SPEED\_8 PO\_8030\_BAYER\_CLOCK\_8

### 7.31.3 Function Documentation

7.31.3.1 int e\_po8030d\_config\_cam ( unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode* )

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

#### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2. Moreover greyscale is twice faster than color mode.

## Parameters

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

## Returns

Zero if the settings are correct, non-zero if an error occur

7.31.3.2 `int e_po8030d_get_bytes_per_pixel ( int color_mode )`

Return the number of bytes per pixel in the given color mode

## Parameters

<i>color_mode</i>	The given color mode
-------------------	----------------------

## Returns

The number of bytes per pixel in the given color mode

7.31.3.3 `void e_po8030d_set_awb_ae ( int awb, int ae )`

Enable/Disable AWB and AE

## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

7.31.3.4 `void e_po8030d_set_bank ( unsigned char bank )`

Set the camera register bank to use

## Parameters

<i>bank</i>	The bank used.
-------------	----------------

See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.31.3.5 void e\_po8030d\_set\_bayer\_clkdiv ( unsigned char *div* )

Set the clock divider

Parameters

<i>div</i>	the register value
------------	--------------------

See also

[BAYER\\_CLOCK\\_1](#), [BAYER\\_CLOCK\\_2](#), [BAYER\\_CLOCK\\_4](#), [BAYER\\_CLOCK\\_8](#)

7.31.3.6 void e\_po8030d\_set\_brightness ( signed char *value* )

Set brightness

Parameters

<i>value</i>	Brightness => [7]:[6:0] = Sign:Magnitude: luminance = Y*contrast + brightness
--------------	---

7.31.3.7 void e\_po8030d\_set\_color\_mode ( unsigned char *format* )

7.31.3.8 void e\_po8030d\_set\_exposure ( unsigned long *exp* )

Set exposure time

Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

7.31.3.9 void e\_po8030d\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

Parameters

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

7.31.3.10 `int e_po8030d_set_mode ( unsigned char format, unsigned char sampl_mode )`

Set camera sampling and color mode

Parameters

<i>format</i>	The color format
<i>sampl_mode</i>	The sampling mode

See also

[GREY\\_SCALE\\_MODE](#), [RGB\\_565\\_MODE](#), [YUV\\_MODE](#), [MODE\\_VGA](#), [MODE\\_QVGA](#), [MODE\\_QQVGA](#)

7.31.3.11 `void e_po8030d_set_pclkdiv ( unsigned char div )`

Set the pixel clock divider

Parameters

<i>div</i>	the register value
------------	--------------------

Warning

The register value is dependant of the camera color and sample mode

7.31.3.12 `void e_po8030d_set_rgb_gain ( unsigned char r, unsigned char g, unsigned char b )`

7.31.3.13 `void e_po8030d_set_sketch_mode ( int mode )`

Set the camera sketch mode (aka, sobel-filter)

Parameters

<i>mode</i>	The sketch mode
-------------	-----------------

See also

[E\\_PO8030D\\_SKETCH\\_BW](#), [E\\_PO8030D\\_SKETCH\\_COLOR](#)

7.31.3.14 `int e_po8030d_set_vsync ( unsigned int start, unsigned int stop )`

Set the camera window VSYNC coordinate

## Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row

## Returns

Zero if OK, non-zero if an error occur

7.31.3.15 int e\_po8030d\_set\_wx ( unsigned int *start*, unsigned int *stop* )

Set the camera window X coordinate

## Parameters

<i>start</i>	The start column
<i>stop</i>	The stop column

## Returns

Zero if OK, non-zero if an error occur

7.31.3.16 int e\_po8030d\_set\_wy ( unsigned int *start*, unsigned int *stop* )

Set the camera window Y coordinate

## Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row

## Returns

Zero if OK, non-zero if an error occur

7.31.3.17 void e\_po8030d\_write\_register ( unsigned char *bank*, unsigned char *reg*, unsigned char *value* )

Set the register reg to value

## Parameters

<i>bank</i>	The register bank
<i>reg</i>	The register address
<i>value</i>	The value to write

See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.31.3.18 int `init_po8030` ( )

7.31.3.19 int `testWriteReg` ( unsigned char *bank* )

## 7.32 camera/fast\_2\_timer/e\_po8030d\_registers.c File Reference

Manage po8030d registers (two timers)

```
#include "../..//I2C/e_I2C_protocol.h"
#include "e_po8030d.h"
```

### Macros

- #define `MCLK` ((long) 14745600) /\* 14.7456Mhz \*/
- #define `MCLK_P_NS` 0.067816840278 /\* Master clock period in ns \*/
- #define `ARRAY_ORIGINE_X` 68
- #define `ARRAY_ORIGINE_Y` 4
- #define `DEVICE_ID` 0xDC
- #define `BANK_REGISTER` 0x3

### Functions

- void `e_po8030d_set_bank` (unsigned char bank)
- void `e_po8030d_write_register` (unsigned char bank, unsigned char reg, unsigned char value)
- unsigned char `e_po8030d_read_register` (unsigned char bank, unsigned char reg)
- int `testWriteReg` (unsigned char bank)
- int `enablePO8030` ()
- int `init_po8030` ()
- void `e_po8030d_set_bayer_clkdiv` (unsigned char div)
- void `e_po8030d_set_pclkdiv` (unsigned char div)
- static int `e_po8030d_set_sampl_gray` (unsigned char sample)
- static int `e_po8030d_set_sampl_color` (unsigned char sample)
- int `e_po8030d_set_mode` (unsigned char format, unsigned char sampl\_mode)
- int `e_po8030d_set_wx` (unsigned int start, unsigned int stop)
- int `e_po8030d_set_wy` (unsigned int start, unsigned int stop)
- int `e_po8030d_set_vsync` (unsigned int start, unsigned int stop)
- void `e_po8030d_set_sketch_mode` (int mode)
- void `e_po8030d_set_mirror` (int vertical, int horizontal)
- void `e_po8030d_set_awb_ae` (int awb, int ae)
- void `e_po8030d_set_rgb_gain` (unsigned char r, unsigned char g, unsigned char b)
- void `e_po8030d_set_exposure` (unsigned long exp)
- void `e_po8030d_set_brightness` (signed char value)



### 7.32.1 Detailed Description

Manage po8030d registers (two timers)

Author

Stefano Morgani

### 7.32.2 Macro Definition Documentation

7.32.2.1 `#define ARRAY_ORIGINE_X 68`

7.32.2.2 `#define ARRAY_ORIGINE_Y 4`

7.32.2.3 `#define BANK_REGISTER 0x3`

7.32.2.4 `#define DEVICE_ID 0xDC`

7.32.2.5 `#define MCLK ((long) 14745600) /* 14.7456Mhz */`

7.32.2.6 `#define MCLK_P_NS 0.067816840278 /* Master clock period in ns */`

### 7.32.3 Function Documentation

7.32.3.1 `unsigned char e_po8030d_read_register ( unsigned char bank, unsigned char reg )`

Read the register *reg*.

Parameters

<i>bank</i>	The register bank
<i>reg</i>	The register address

Returns

The register value

See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.32.3.2 `void e_po8030d_set_awb_ae ( int awb, int ae )`

Enable/Disable AWB and AE

## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

7.32.3.3 void e\_po8030d\_set\_bank ( unsigned char *bank* )

Set the camera register bank to use

## Parameters

<i>bank</i>	The bank used.
-------------	----------------

## See also

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.32.3.4 void e\_po8030d\_set\_bayer\_clkdiv ( unsigned char *div* )

Set the clock divider

## Parameters

<i>div</i>	the register value
------------	--------------------

## See also

[BAYER\\_CLOCK\\_1](#), [BAYER\\_CLOCK\\_2](#), [BAYER\\_CLOCK\\_4](#), [BAYER\\_CLOCK\\_8](#)

7.32.3.5 void e\_po8030d\_set\_brightness ( signed char *value* )

Set brightness

## Parameters

<i>value</i>	Brightness => [7]:[6:0] = Sign:Magnitude: luminance = Y*contrast + brightness
--------------	---

7.32.3.6 void e\_po8030d\_set\_exposure ( unsigned long *exp* )

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

7.32.3.7 void e\_po8030d\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

Parameters

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

7.32.3.8 int e\_po8030d\_set\_mode ( unsigned char *format*, unsigned char *sampl\_mode* )

Set camera sampling and color mode

Parameters

<i>format</i>	The color format
<i>sampl_mode</i>	The sampling mode

See also

[GREY\\_SCALE\\_MODE](#), [RGB\\_565\\_MODE](#), [YUV\\_MODE](#), [MODE\\_VGA](#), [MODE\\_QVGA](#), [MODE\\_QQVGA](#)

7.32.3.9 void e\_po8030d\_set\_pclkdiv ( unsigned char *div* )

Set the pixel clock divider

Parameters

<i>div</i>	the register value
------------	--------------------

Warning

The register value is dependant of the camera color and sample mode

7.32.3.10 void e\_po8030d\_set\_rgb\_gain ( unsigned char *r*, unsigned char *g*, unsigned char *b* )

7.32.3.11 static int e\_po8030d\_set\_sampl\_color ( unsigned char *sample* ) [static]

7.32.3.12 static int e\_po8030d\_set\_sampl\_gray ( unsigned char *sample* ) [static]

7.32.3.13 void e\_po8030d\_set\_sketch\_mode ( int *mode* )

Set the camera sketch mode (aka, sobel-filter)

**Parameters**

<i>mode</i>	The sketch mode
-------------	-----------------

**See also**

[E\\_PO8030D\\_SKETCH\\_BW](#), [E\\_PO8030D\\_SKETCH\\_COLOR](#)

7.32.3.14 `int e_po8030d_set_vsync ( unsigned int start, unsigned int stop )`

Set the camera window VSYNC coordinate

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row

**Returns**

Zero if OK, non-zero if an error occur

7.32.3.15 `int e_po8030d_set_wx ( unsigned int start, unsigned int stop )`

Set the camera window X coordinate

**Parameters**

<i>start</i>	The start column
<i>stop</i>	The stop column

**Returns**

Zero if OK, non-zero if an error occur

7.32.3.16 `int e_po8030d_set_wy ( unsigned int start, unsigned int stop )`

Set the camera window Y coordinate

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row

**Returns**

Zero if OK, non-zero if an error occur

7.32.3.17 void e\_po8030d\_write\_register ( unsigned char *bank*, unsigned char *reg*, unsigned char *value* )

Set the register reg to value

**Parameters**

<i>bank</i>	The register bank
<i>reg</i>	The register address
<i>value</i>	The value to write

**See also**

[BANK\\_A](#), [BANK\\_B](#), [BANK\\_C](#), [BANK\\_D](#)

7.32.3.18 int enablePO8030 ( )

7.32.3.19 int init\_po8030 ( )

7.32.3.20 int testWriteReg ( unsigned char *bank* )

**7.33 camera/fast\_2\_timer/e\_poxxxx.h File Reference**

Camera library header.

**Macros**

- #define [POXXXX\\_FULL](#) 1
- #define [ARRAY\\_WIDTH](#) 640
- #define [ARRAY\\_HEIGHT](#) 480
- #define [GREY\\_SCALE\\_MODE](#) 0
- #define [RGB\\_565\\_MODE](#) 1
- #define [YUV\\_MODE](#) 2

**Functions**

- int [e\\_poxxxx\\_config\\_cam](#) (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- int [e\\_poxxxx\\_init\\_cam](#) (void)
- int [e\\_poxxxx\\_get\\_orientation](#) (void)
- void [e\\_poxxxx\\_write\\_cam\\_registers](#) (void)
- void [e\\_poxxxx\\_launch\\_capture](#) (char \*buf)
- int [e\\_poxxxx\\_is\\_img\\_ready](#) (void)
- void [e\\_poxxxx\\_set\\_mirror](#) (int vertical, int horizontal)
- int [e\\_poxxxx\\_apply\\_timer\\_config](#) (int pixel\_row, int pixel\_col, int bpp, int pbp, int bbl)
- void [e\\_poxxxx\\_set\\_awb\\_ae](#) (int awb, int ae)
- void [e\\_poxxxx\\_set\\_rgb\\_gain](#) (unsigned char r, unsigned char g, unsigned char b)
- void [e\\_poxxxx\\_set\\_exposure](#) (unsigned long exp)

### 7.33.1 Detailed Description

Camera library header.

Author

Philippe Rétornaz

### 7.33.2 Macro Definition Documentation

7.33.2.1 `#define ARRAY_HEIGHT 480`

7.33.2.2 `#define ARRAY_WIDTH 640`

7.33.2.3 `#define GREY_SCALE_MODE 0`

7.33.2.4 `#define POXXXX_FULL 1`

If you set this at 0, you save about 168 bytes of memory But you loose all advanced camera functions

7.33.2.5 `#define RGB_565_MODE 1`

7.33.2.6 `#define YUV_MODE 2`

### 7.33.3 Function Documentation

7.33.3.1 `int e_poxxxx_apply_timer_config ( int pixel_row, int pixel_col, int bpp, int pbp, int bbl )`

Modify the interrupt configuration

Warning

This is an internal function, use `e_poxxxx_config_cam`

Parameters

<i>pixel_row</i>	The number of row to take
<i>pixel_col</i>	The number of pixel to take each <i>pixel_row</i>
<i>bpp</i>	The number of byte per pixel
<i>pbp</i>	The number of pixel to ignore between each pixel
<i>bbl</i>	The number of row to ignore between each line

Returns

Zero if OK, non-zero if the mode exceed internal data representation

See also

[e\\_poxxxx\\_config\\_cam](#)

7.33.3.2 `int e_poxxxx_config_cam ( unsigned int sensor_x1, unsigned int sensor_y1, unsigned int sensor_width, unsigned int sensor_height, unsigned int zoom_fact_width, unsigned int zoom_fact_height, int color_mode )`

7.33.3.3 `int e_poxxxx_get_orientation ( void )`

Return the camera orientation

Returns

0: vertical 480x640, 1: horizontal 640x480, horizontal inverted, -1: unknown

7.33.3.4 `int e_poxxxx_init_cam ( void )`

Initialize the camera, return the version in hexa, 0x3030 or 0x6030

7.33.3.5 `int e_poxxxx_is_img_ready ( void )`

Check if the current capture is finished

Returns

Zero if the current capture is in progress, non-zero if the capture is done.

See also

[e\\_poxxxx\\_launch\\_capture](#)

7.33.3.6 `void e_poxxxx_launch_capture ( char * buf )`

Launch a capture in the *buf* buffer

Parameters

<i>buf</i>	The buffer to write to
------------	------------------------

See also

[e\\_poxxxx\\_config\\_cam](#) and [e\\_poxxxx\\_is\\_img\\_ready](#)

7.33.3.7 void e\_poxxxx\_set\_awb\_ae ( int *awb*, int *ae* )

Enable/Disable AWB and AE



## Parameters

<i>awb</i>	1 means AWB enabled, 0 means disabled
<i>ae</i>	1 means AE enabled, 0 means disabled

7.33.3.8 void e\_poxxxx\_set\_exposure ( unsigned long *exp* )

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

## Warning

Only meaningful if AE is disabled

7.33.3.9 void e\_poxxxx\_set\_mirror ( int *vertical*, int *horizontal* )7.33.3.10 void e\_poxxxx\_set\_rgb\_gain ( unsigned char *r*, unsigned char *g*, unsigned char *b* )

Set the gains of the camera

## Parameters

<i>red</i>	The red pixels' gain (0..255)
<i>green</i>	The green pixels' gain (0..255)
<i>blue</i>	The blue pixels' gain (0..255)

## Warning

Only meaningful if AWB is disabled

## 7.33.3.11 void e\_poxxxx\_write\_cam\_registers ( void )

## 7.34 camera/fast\_2\_timer/e\_timers.c File Reference

Manage camera's interrupts (two timers)

```
#include <p30F6014A.h>
#include "../motor_led/e_epuck_ports.h"
#include "e_po3030k.h"
```

## Functions

- int `__attribute__` ((near))
- static void `init_timer5` (void)
- static void `init_timer4` (void)
- void `e_poxxxx_launch_capture` (char \*buf)
- int `e_poxxxx_apply_timer_config` (int pixel\_row, int pixel\_col, int bpp, int pbp, int bbl)
- int `e_poxxxx_is_img_ready` (void)

## Variables

- char \* `_poxxxx_buffer`
- int `_poxxxx_img_ready`
- static int `blank_row_betw`

### 7.34.1 Detailed Description

Manage camera's interrupts (two timers)

#### Author

Philippe Rétornaz

### 7.34.2 Function Documentation

7.34.2.1 int `__attribute__` ( (near) )

7.34.2.2 int `e_poxxxx_apply_timer_config` ( int *pixel\_row*, int *pixel\_col*, int *bpp*, int *pbp*, int *bbl* )

Modify the interrupt configuration

#### Warning

This is an internal function, use `e_poxxxx_config_cam`

#### Parameters

<i>pixel_row</i>	The number of row to take
<i>pixel_col</i>	The number of pixel to take each <i>pixel_row</i>
<i>bpp</i>	The number of byte per pixel
<i>pbp</i>	The number of pixel to ignore between each pixel
<i>bbl</i>	The number of row to ignore between each line

**Returns**

Zero if OK, non-zero if the mode exceed internal data representation

**See also**

[e\\_poxxxx\\_config\\_cam](#)

**7.34.2.3 int e\_poxxxx\_is\_img\_ready ( void )**

Check if the current capture is finished

**Returns**

Zero if the current capture is in progress, non-zero if the capture is done.

**See also**

[e\\_poxxxx\\_launch\\_capture](#)

**7.34.2.4 void e\_poxxxx\_launch\_capture ( char \* buf )**

Launch a capture in the *buf* buffer

**Parameters**

<i>buf</i>	The buffer to write to
------------	------------------------

**See also**

[e\\_poxxxx\\_config\\_cam](#) and [e\\_poxxxx\\_is\\_img\\_ready](#)

**7.34.2.5 static void init\_timer4 ( void ) [static]****7.34.2.6 static void init\_timer5 ( void ) [static]****7.34.3 Variable Documentation****7.34.3.1 char\* \_poxxxx\_buffer**

The buffer to write to

### 7.34.3.2 `int _poxxxx_img_ready`

The flag to tell, the image is ready or not Zero mean capture is in progress, non-zero mean capture done.

See also

[e\\_po3030k\\_is\\_img\\_ready](#)

### 7.34.3.3 `int blank_row_betw` [static]

## 7.35 `camera/slow_3_timer/e_timers.c` File Reference

Manage camera's interrupts (three timers)

```
#include <p30f6014a.h>
#include "../motor_LED/e_epuck_ports.h"
#include "e_po3030k.h"
```

### Functions

- void [\\_\\_attribute\\_\\_](#) ((interrupt, auto\_psv))
  - The HSYNC interrupt. This interrupt is called each time the Horizontal sync signal is asserted This mean we begin the a line of the picture.*
- static void [init\\_timer5](#) (void)
- static void [init\\_timer4](#) (void)
- static void [init\\_timer1](#) (void)
- void [e\\_po3030k\\_launch\\_capture](#) (char \*buf)
- void [e\\_po3030k\\_apply\\_timer\\_config](#) (int pixel\_row, int pixel\_col, int bpp, int pbp, int bbl)
- int [e\\_po3030k\\_is\\_img\\_ready](#) (void)

### Variables

- static int [max\\_row](#)
- static int [max\\_col](#)
- static char \* [buffer](#)
- static int [img\\_ready](#)
- static int [current\\_row](#)
- static int [current\\_col](#)
- static int [buf\\_pos](#)
- static int [bytes\\_per\\_pixel](#)
- static int [bpp\\_current](#)
- static int [pixel\\_betw\\_pixel](#)
- static int [pbp\\_current](#)
- static int [blank\\_betw\\_lines](#)
- static int [bbl\\_current](#)

### 7.35.1 Detailed Description

Manage camera's interrupts (three timers)

### 7.35.2 Function Documentation

#### 7.35.2.1 void \_\_attribute\_\_((interrupt, auto\_psv))

The HSYNC interrupt. This interrupt is called each time the Horizontal sync signal is asserted This mean we begin the a line of the picture.

The Pixel Clock interrupt. This interrupt is called every time the Pixel clock signal is asserted This mean that the next byte is ready to be read.

The VSYNC interrupt. This interrupt is called every time the Vertical sync signal is asserted This mean that the picture is comming from the camera ( we will have the first line soon )

#### 7.35.2.2 void e\_po3030k\_apply\_timer\_config ( int *pixel\_row*, int *pixel\_col*, int *bpp*, int *pbp*, int *bbl* )

Modify the interrupt configuration

#### Warning

This is an internal function, use `e_po3030k_config_cam`

#### Parameters

<i>pixel_row</i>	The number of row to take
<i>pixel_col</i>	The number of pixel to take each <i>pixel_row</i>
<i>bpp</i>	The number of byte per pixel
<i>pbp</i>	The number of pixel to ignore between each pixel
<i>bbl</i>	The number of row to ignore between each line

#### See also

[e\\_po3030k\\_get\\_bytes\\_per\\_pixel](#) and [e\\_po3030k\\_config\\_cam](#)

#### 7.35.2.3 int e\_po3030k\_is\_img\_ready ( void )

Check if the current capture is finished

#### Returns

Zero if the current capture is in progress, non-zero if the capture is done.

#### See also

[e\\_po3030k\\_launch\\_capture](#)

#### 7.35.2.4 void e\_po3030k\_launch\_capture ( char \* buf )

Launch a capture in the *buf* buffer

##### Parameters

<i>buf</i>	The buffer to write to
------------	------------------------

##### See also

[e\\_po3030k\\_config\\_cam](#) and [e\\_po3030k\\_is\\_img\\_ready](#)

#### 7.35.2.5 static void init\_timer1 ( void ) [static]

#### 7.35.2.6 static void init\_timer4 ( void ) [static]

#### 7.35.2.7 static void init\_timer5 ( void ) [static]

### 7.35.3 Variable Documentation

#### 7.35.3.1 int bbl\_current [static]

The current row we ignore between each effective row

#### 7.35.3.2 int blank\_betw\_lines [static]

The number of blank row between each effective row

#### 7.35.3.3 int bpp\_current [static]

The current byte we are inside a pixel

#### 7.35.3.4 int buf\_pos [static]

Counter which is incremented each time we acquire a byte

#### 7.35.3.5 char\* buffer [static]

The buffer to write to

#### 7.35.3.6 int bytes\_per\_pixel [static]

Number of bytes per pixel

### 7.35.3.7 int current\_col [static]

The current column we are

### 7.35.3.8 int current\_row [static]

The current row we are

### 7.35.3.9 int img\_ready [static]

The flag to tell, the image is ready or not Zero mean capture is in progress, non-zero mean capture done.

See also

[e\\_po3030k\\_is\\_img\\_ready](#)

### 7.35.3.10 int max\_col [static]

The number of bytes per row we should take

See also

[e\\_po3030k\\_get\\_bytes\\_per\\_pixel](#), [e\\_po3030k\\_apply\\_timer\\_config](#)

### 7.35.3.11 int max\_row [static]

The number of row we should take

See also

[e\\_po3030k\\_aApply\\_timer\\_config](#)

### 7.35.3.12 int pbp\_current [static]

The current pixel we are between two effective pixel

### 7.35.3.13 int pixel\_betw\_pixel [static]

Pixel to "jump" between each effective pixel

## 7.36 camera/slow\_3\_timer/e\_calc.c File Reference

Calculate the timing for the camera (three timers)

```
#include "e_po3030k.h"
#include "../motor_LED/e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../motor_LED/e_init_port.h"
```

### Macros

- #define [ARRAY\\_ORIGINE\\_X](#) 210
- #define [ARRAY\\_ORIGINE\\_Y](#) 7

### Functions

- int [e\\_po3030k\\_config\\_cam](#) (unsigned int sensor\_x1, unsigned int sensor\_y1, unsigned int sensor\_width, unsigned int sensor\_height, unsigned int zoom\_fact\_width, unsigned int zoom\_fact\_height, int color\_mode)
- int [e\\_po3030k\\_get\\_bytes\\_per\\_pixel](#) (int color\_mode)
- void [e\\_po3030k\\_init\\_cam](#) (void)

### 7.36.1 Detailed Description

Calculate the timing for the camera (three timers)

### 7.36.2 Macro Definition Documentation

7.36.2.1 #define [ARRAY\\_ORIGINE\\_X](#) 210

7.36.2.2 #define [ARRAY\\_ORIGINE\\_Y](#) 7

### 7.36.3 Function Documentation

7.36.3.1 int [e\\_po3030k\\_config\\_cam](#) ( unsigned int *sensor\_x1*, unsigned int *sensor\_y1*, unsigned int *sensor\_width*, unsigned int *sensor\_height*, unsigned int *zoom\_fact\_width*, unsigned int *zoom\_fact\_height*, int *color\_mode* )

This function setup the internal timing of the camera to match the zoom, color mode and interest area.

#### Warning

If the common denominator of both zoom factor is 4 or 2, part of the subsampling is done by the camera ( QQVGA = 4, QVGA = 2 ). This increase the framerate by respectively 4 or 2.



## Parameters

<i>sensor_x1</i>	The X coordinate of the window's corner
<i>sensor_y1</i>	The Y coordinate of the window's corner
<i>sensor_width</i>	the Width of the interest area, in FULL sampling scale
<i>sensor_height</i>	The Height of the interest area, in FULL sampling scale
<i>zoom_fact_width</i>	The subsampling to apply for the window's Width
<i>zoom_fact_height</i>	The subsampling to apply for the window's Height
<i>color_mode</i>	The color mode in which the camera should be configured

## Returns

Zero if the settings are correct, non-zero if an error occur

## See also

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.36.3.2 int e\_po3030k\_get\_bytes\_per\_pixel ( int *color\_mode* )

Return the number of bytes per pixel in the given color mode

## Parameters

<i>color_mode</i>	The given color mode
-------------------	----------------------

## Returns

The number of bytes per pixel in the given color mode

## 7.36.3.3 void e\_po3030k\_init\_cam ( void )

Initialize the camera, must be called before any other function

## 7.37 camera/slow\_3\_timer/e\_registers.c File Reference

Manage po3030k registers (three timers)

```
#include "../I2C/e_I2C_protocol.h"
#include "e_po3030k.h"
```

## Macros

- #define MCLK ((long) 14745600) /\* 14.7456Mhz \*/
- #define MCLK\_P\_NS 0.067816840278 /\* Master clock period in ns \*/
- #define ARRAY\_ORIGINE\_X 210
- #define ARRAY\_ORIGINE\_Y 7
- #define BASE\_D1 0
- #define BASE\_D2 0
- #define BASE\_D3 0
- #define BASE\_D4 0
- #define DEVICE\_ID 0xDC
- #define FRAME\_WIDTH 0x0353
- #define FRAME\_HEIGHT 0x01e8
- #define WINDOW\_X1\_BASE 9
- #define WINDOW\_Y1\_BASE 13
- #define WINDOW\_X2\_BASE 17
- #define WINDOW\_Y2\_BASE 21
- #define BIAS\_BASE 25
- #define COLGAIN\_BASE 29
- #define INTEGR\_BASE 39
- #define SPEED\_ADDR (45 - BASE\_D1)
- #define MIRROR\_BASE 47
- #define SAMPLING\_ADDR ( 51 - BASE\_D2 )
- #define ADCOFF\_BASE ( 53 - BASE\_D2 )
- #define FLICKM\_BASE ( 55 - BASE\_D2 )
- #define FLICKP\_BASE 59
- #define COLOR\_M\_ADDR (67 - BASE\_D3)
- #define MODE\_R5G6B5 0x08
- #define MODE\_YUV 0x02
- #define MODE\_GRAYSCALE 0x0c
- #define SEPIA\_BASE 69
- #define LENS\_G\_BASE 73
- #define EDGE\_BASE 79
- #define GAMMA\_BASE 83
- #define COLOR\_COEF\_BASE 107
- #define CBCRGAIN\_BASE 125
- #define BRICTR\_BASE 129
- #define SEPIATONE\_BASE 133
- #define VSYNCSTART\_BASE (145 - BASE\_D4)
- #define VSYNCSTOP\_BASE (149 - BASE\_D4)
- #define VSYNCCOL\_BASE (153 - BASE\_D4)
- #define WW\_BASE 157
- #define AWVAETOL\_BASE 159
- #define AESPEED\_BASE 161
- #define EXPOSURE\_BASE 163
- #define REFREPO\_BASE 169
- #define MINMAXEXP\_BASE 175
- #define MINMAXAWB\_BASE 183
- #define WEIGHWIN\_BASE 195
- #define AWBAEENABLE\_BASE 211
- #define GAMMASELCOL\_BASE 215
- #define NB\_REGISTERS (sizeof(cam\_reg)/(2\*sizeof(cam\_reg[0])))

## Functions

- void [e\\_po3030k\\_write\\_cam\\_registers](#) (void)
- void [e\\_po3030k\\_read\\_cam\\_registers](#) (void)
- int [e\\_po3030k\\_set\\_color\\_mode](#) (int mode)
- int [e\\_po3030k\\_set\\_sampling\\_mode](#) (int mode)
- int [e\\_po3030k\\_set\\_speed](#) (int mode)
- int [e\\_po3030k\\_set\\_wx](#) (unsigned int start, unsigned int stop)
- int [e\\_po3030k\\_set\\_wy](#) (unsigned int start, unsigned int stop)
- int [e\\_po3030k\\_set\\_vsync](#) (unsigned int start, unsigned int stop, unsigned int col)
- int [e\\_po3030k\\_set\\_register](#) (unsigned char adr, unsigned char value)
- int [e\\_po3030k\\_get\\_register](#) (unsigned char adr, unsigned char \*value)
- void [e\\_po3030k\\_set\\_bias](#) (unsigned char pixbias, unsigned char opbias)
- int [e\\_po3030k\\_set\\_color\\_gain](#) (unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue)
- void [e\\_po3030k\\_set\\_integr\\_time](#) (unsigned long time)
- void [e\\_po3030k\\_set\\_mirror](#) (int vertical, int horizontal)
- void [e\\_po3030k\\_set\\_adc\\_offset](#) (unsigned char offset)
- void [e\\_po3030k\\_set\\_sepia](#) (int status)
- void [e\\_po3030k\\_set\\_lens\\_gain](#) (unsigned char red, unsigned char green, unsigned char blue)
- void [e\\_po3030k\\_set\\_edge\\_prop](#) (unsigned char gain, unsigned char tresh)
- void [e\\_po3030k\\_set\\_gamma\\_coef](#) (unsigned char array[12], char color)
- void [e\\_po3030k\\_write\\_gamma\\_coef](#) (void)
- int [e\\_po3030k\\_sync\\_register\\_array](#) (unsigned char start, unsigned char stop)
- void [e\\_po3030k\\_SetColorMatrix](#) (unsigned char array[3 \* 3])
- void [e\\_po3030k\\_set\\_cb\\_cr\\_gain](#) (unsigned char cg11c, unsigned char cg22c)
- void [e\\_po3030k\\_set\\_brigh\\_contr](#) (unsigned char bright, unsigned char contrast)
- void [e\\_po3030k\\_set\\_sepia\\_tone](#) (unsigned char cb, unsigned char cr)
- void [e\\_po3030k\\_set\\_ww](#) (unsigned char ww)
- void [e\\_po3030k\\_set\\_awb\\_ae\\_tol](#) (unsigned char awbm, unsigned char aem)
- void [e\\_po3030k\\_set\\_ae\\_speed](#) (unsigned char b, unsigned char d)
- void [e\\_po3030k\\_set\\_exposure](#) (long t)
- void [e\\_po3030k\\_set\\_ref\\_exposure](#) (unsigned char exp)
- void [e\\_po3030k\\_set\\_max\\_min\\_exp](#) (unsigned int max, unsigned int min)
- void [e\\_po3030k\\_set\\_max\\_min\\_awb](#) (unsigned char minb, unsigned char maxb, unsigned char minr, unsigned char maxr, unsigned char ratiob, unsigned char ratiob)
- int [e\\_po3030k\\_set\\_weight\\_win](#) (unsigned int x1, unsigned int x2, unsigned int y1, unsigned int y2)
- void [e\\_po3030k\\_set\\_awb\\_ae](#) (int awb, int ae)
- void [e\\_po3030k\\_set\\_flicker\\_mode](#) (int manual)
- void [e\\_po3030k\\_set\\_flicker\\_detection](#) (int hz50, int hz60)
- static long [po3030k\\_get\\_pixelclock](#) (void)
- int [e\\_po3030k\\_set\\_flicker\\_man\\_set](#) (int hz50, int hz60, int fdm, int fk, int tol)

## Variables

- static unsigned char [cam\\_reg](#) []

### 7.37.1 Detailed Description

Manage po3030k registers (three timers)

#### Author

Philippe Rétornaz

## 7.37.2 Macro Definition Documentation

7.37.2.1 #define ADCOFF\_BASE ( 53 - BASE\_D2 )

7.37.2.2 #define AESPEED\_BASE 161

7.37.2.3 #define ARRAY\_ORIGINE\_X 210

7.37.2.4 #define ARRAY\_ORIGINE\_Y 7

7.37.2.5 #define AWBAEENABLE\_BASE 211

7.37.2.6 #define AWVAETOL\_BASE 159

7.37.2.7 #define BASE\_D1 0

7.37.2.8 #define BASE\_D2 0

7.37.2.9 #define BASE\_D3 0

7.37.2.10 #define BASE\_D4 0

7.37.2.11 #define BIAS\_BASE 25

7.37.2.12 #define BRICTR\_BASE 129

7.37.2.13 #define CBCRGAIN\_BASE 125

7.37.2.14 #define COLGAIN\_BASE 29

7.37.2.15 #define COLOR\_COEF\_BASE 107

7.37.2.16 #define COLOR\_M\_ADDR (67 - BASE\_D3)

7.37.2.17 #define DEVICE\_ID 0xDC

7.37.2.18 #define EDGE\_BASE 79

7.37.2.19 #define EXPOSURE\_BASE 163

7.37.2.20 #define FLICKM\_BASE ( 55 - BASE\_D2 )

7.37.2.21 #define FLICKP\_BASE 59

7.37.2.22 #define FRAME\_HEIGHT 0x01e8

7.37.2.23 #define FRAME\_WIDTH 0x0353

7.37.2.24 #define GAMMA\_BASE 83

7.37.2.25 #define GAMMASELCOL\_BASE 215

7.37.2.26 #define INTEGR\_BASE 39

7.37.2.27 #define LENS\_G\_BASE 73

7.37.2.28 #define MCLK (((long) 14745600) /\* 14.7456Mhz \*/)

7.37.2.29 #define MCLK\_P\_NS 0.067816840278 /\* Master clock period in ns \*/

7.37.2.30 #define MINMAXAWB\_BASE 183

7.37.2.31 #define MINMAXEXP\_BASE 175

7.37.2.32 #define MIRROR\_BASE 47

7.37.2.33 #define MODE\_GRAYSCALE 0x0c

7.37.2.34 #define MODE\_R5G6B5 0x08

7.37.2.35 #define MODE\_YUV 0x02

7.37.2.36 #define NB\_REGISTERS (sizeof(cam\_reg)/(2\*sizeof(cam\_reg[0])))

7.37.2.37 #define REFREPO\_BASE 169

7.37.2.38 #define SAMPLING\_ADDR (51 - BASE\_D2)

7.37.2.39 #define SEPIA\_BASE 69

7.37.2.40 #define SEPIATONE\_BASE 133

7.37.2.41 #define SPEED\_ADDR (45 - BASE\_D1)

7.37.2.42 #define VSYNCCOL\_BASE (153 - BASE\_D4)

7.37.2.43 #define VSYNCSTART\_BASE (145 - BASE\_D4)

7.37.2.44 #define VSYNCSTOP\_BASE (149 - BASE\_D4)

7.37.2.45 #define WEIGHWIN\_BASE 195

7.37.2.46 `#define WINDOW_X1_BASE 9`

7.37.2.47 `#define WINDOW_X2_BASE 17`

7.37.2.48 `#define WINDOW_Y1_BASE 13`

7.37.2.49 `#define WINDOW_Y2_BASE 21`

7.37.2.50 `#define WW_BASE 157`

### 7.37.3 Function Documentation

7.37.3.1 `int e_po3030k_get_register ( unsigned char adr, unsigned char * value )`

Get the register *adr* value

#### Parameters

<i>adr</i>	The address
<i>value</i>	The pointer to the value to write to

#### Returns

Zero if register found, non-zero if not found

#### Warning

This function is sub-optimal, if you use it heavily add an internal function to register.c

#### See also

[e\\_po3030k\\_set\\_register](#)

7.37.3.2 `void e_po3030k_read_cam_registers ( void )`

Read the camera register

#### See also

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.37.3.3 `void e_po3030k_set_adc_offset ( unsigned char offset )`

Set the Analog to Digital Converter offset

## Parameters

<i>offset</i>	The offset
---------------	------------

## See also

Datasheet p.28

7.37.3.4 void e\_po3030k\_set\_ae\_speed ( unsigned char *b*, unsigned char *d* )

Set AE speed

## Parameters

<i>b</i>	AE speed factor when exposure time is decreasing ( 4 lower bits only )
<i>d</i>	AE speed factor when exposure time is increasing ( 4 lower bits only )

## See also

Datasheet p.44

7.37.3.5 void e\_po3030k\_set\_awb\_ae ( int *awb*, int *ae* )

Enable/Disable AWB and AE

## Parameters

<i>awb</i>	1 mean AWB enabled, 0 mean disabled
<i>ae</i>	1 mean AE enabled, 0 mean disabled

## See also

Datasheet p. 50

7.37.3.6 void e\_po3030k\_set\_awb\_ae\_tol ( unsigned char *awbm*, unsigned char *aem* )

Set AWB/AE tolerance margin

## Parameters

<i>awbm</i>	AWV Margin ( 4 lower bits only )
<i>aem</i>	AW Margin ( 4 lower bits only )

**See also**

Datasheet p.44

### 7.37.3.7 void e\_po3030k\_set\_bias ( unsigned char *pixbias*, unsigned char *opbias* )

Set the Pixel and amplifcator bias Increasing the bias produce better image quality, but increase camera's power consumption

**Parameters**

<i>pixbias</i>	The pixel bias
<i>opbias</i>	The Amplificator bias

**See also**

Datasheet p.22

### 7.37.3.8 void e\_po3030k\_set\_brigh\_contr ( unsigned char *bright*, unsigned char *contrast* )

Set the Brightness & Contrast

**Parameters**

<i>bright</i>	The Brightness ( signed 1+7bits fixed point format )
<i>contrast</i>	The Contrast

**See also**

Datasheet p. 40

### 7.37.3.9 void e\_po3030k\_set\_cb\_cr\_gain ( unsigned char *cg11c*, unsigned char *cg22c* )

Set The color gain ( Cb/Cr )

**Parameters**

<i>cg11c</i>	Cb gain ( Sign[7]   Integer[6:5]   fractional[4:0] )
<i>cg22c</i>	Cr gain ( Sign[7]   Integer[6:5]   fractional[4:0] )

**See also**

Datasheet p. 40



7.37.3.10 `int e_po3030k_set_color_gain ( unsigned char global, unsigned char red, unsigned char green1, unsigned char green2, unsigned char blue )`

Set the gains of the camera

#### Parameters

<i>global</i>	The global gain $\in \{0, 79\}$
<i>red</i>	The red pixel's gain (fixed point [2:6] format)
<i>green1</i>	The green pixel near read one gain ([2:6] format)
<i>green2</i>	The green pixel near blue one gain ([2:6] format)
<i>blue</i>	The blue pixel's gain ([2:6] format)

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p.23-24

7.37.3.11 `int e_po3030k_set_color_mode ( int mode )`

Set the camera color mode

#### Warning

This is an internal function, use `e_po3030k_config_cam`

#### Parameters

<i>mode</i>	The color mode
-------------	----------------

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p. 31, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

7.37.3.12 `void e_po3030k_set_edge_prop ( unsigned char gain, unsigned char tresh )`

Set Edge properties

## Parameters

<i>gain</i>	Edge gain & moire factor (fixed point [2:3] format)
<i>tresh</i>	Edge Enhancement threshold

## See also

Datasheet p.36

## 7.37.3.13 void e\_po3030k\_set\_exposure ( long t )

Set exposure time

## Parameters

<i>t</i>	Exposure time, LSB is in 1/64 line time
----------	---

## Warning

Only writable if AE is disabled

## See also

Datasheet p.45

## 7.37.3.14 void e\_po3030k\_set\_flicker\_detection ( int hz50, int hz60 )

Set the 50/60Hz flicker detection

## Parameters

<i>hz50</i>	Non-zero mean 50Hz flicker detection enabled (default disabled)
<i>hz60</i>	Non-zero mean 60Hz flicker detection enabled (default disabled)

## Warning

If Automatic mode is enabled and both 50Hz and 60Hz are disabled, camera will enable both. By default, the camera automatically detect 50 and 60Hz flicker.

## See also

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

## 7.37.3.15 int e\_po3030k\_set\_flicker\_man\_set ( int hz50, int hz60, int fdm, int fk, int tol )

Set the camera's manual flicker's detection setting

**Parameters**

<i>hz50</i>	The Hz for the 50Hz detection
<i>hz60</i>	The Hz for the 60Hz detection
<i>fdm</i>	Flicker duration mode
<i>fk</i>	Flicker count step
<i>tol</i>	Flicker tolerance

**Warning**

You must have set the mode ( image size, color ) before calling this function

**Returns**

Non-zero if an error occur, 0 if OK

**See also**

Datasheet p.29-30 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_mode\(\)](#)

**7.37.3.16 void e\_po3030k\_set\_flicker\_mode ( int *manual* )**

Set flicker detection mode

**Parameters**

<i>manual</i>	Non-zero mean manual mode is enabled ( default automatic mode enabled )
---------------	---

**See also**

Datasheet p.29 [e\\_po3030k\\_set\\_flicker\\_detection\(\)](#) [e\\_po3030k\\_set\\_flicker\\_man\\_set\(\)](#)

**7.37.3.17 void e\_po3030k\_set\_gamma\_coef ( unsigned char *array*[12], char *color* )**

Set gamma coefficient

**Warning**

This feature need extra care from the user

**Parameters**

<i>array</i>	Gamma coefficient array
<i>color</i>	First two bytes : - 0b01 => Green <ul style="list-style-type: none"> <li>• 0b00 =&gt; Red</li> <li>• else =&gt; Blue</li> </ul>

## See also

Datasheet p. 38, 50, 57-58 and e\_po3030k\_WriteGammaCoef

7.37.3.18 void e\_po3030k\_set\_integr\_time ( unsigned long *time* )

Set the pixel intergration time This is counted in line-time interval. See dataseet p.25 for more information

## Parameters

<i>time</i>	The integration time ( fixed point [14:6] format )
-------------	--

## See also

Datasheet p.25

7.37.3.19 void e\_po3030k\_set\_lens\_gain ( unsigned char *red*, unsigned char *green*, unsigned char *blue* )

Set lens shading gain

## Parameters

<i>red</i>	Lens gain for red pixel $\in \{0, 15\}$
<i>green</i>	Lens gain for green pixel $\in \{0, 15\}$
<i>blue</i>	Lens gain for blue pixel $\in \{0, 15\}$

## See also

Datasheet p.36

7.37.3.20 void e\_po3030k\_set\_max\_min\_awb ( unsigned char *minb*, unsigned char *maxb*, unsigned char *minr*, unsigned char *maxr*, unsigned char *ratior*, unsigned char *ratiob* )

Set the minimum and maximum red and blue gain in AWB mode

## Parameters

<i>minb</i>	The minimum blue gain
<i>maxb</i>	The maximum blue gain
<i>minr</i>	The minimum red gain
<i>maxr</i>	The maximum red gain
<i>ratior</i>	The red gain ratio
<i>ratiob</i>	The blue gain ratio

**See also**

Datasheet p. 47-48

### 7.37.3.21 void e\_po3030k\_set\_max\_min\_exp ( unsigned int *max*, unsigned int *min* )

Set the minimum and maximum exposure time in AE mode

**Parameters**

<i>min</i>	The minimum exposure time
<i>max</i>	The maximum exposure time

**See also**

Datasheet p.46-47

### 7.37.3.22 void e\_po3030k\_set\_mirror ( int *vertical*, int *horizontal* )

Enable/Disable horizontal or vertical mirror

**Parameters**

<i>vertical</i>	Set to 1 when vertical mirror is enabled, 0 if disabled
<i>horizontal</i>	Set to 1 when horizontal mirror is enabled, 0 if disabled

**See also**

Datasheet p.27

### 7.37.3.23 void e\_po3030k\_set\_ref\_exposure ( unsigned char *exp* )

Set the reference exposure. The average brightness which the AE should have

**Parameters**

<i>exp</i>	The target exposure level
------------	---------------------------

**See also**

Datasheet p.45

### 7.37.3.24 int e\_po3030k\_set\_register ( unsigned char *adr*, unsigned char *value* )

Set the register *adr* to value *value*

**Parameters**

<i>adr</i>	The address
<i>value</i>	The value

**Returns**

Zero if register found, non-zero if not found

**Warning**

This function is sub-optimal, if you use it heavily add an internal function to register.c

**See also**

[e\\_po3030k\\_get\\_register](#)

**7.37.3.25 int e\_po3030k\_set\_sampling\_mode ( int mode )**

Set the camera sampling mode

**Warning**

This is an internal function, use `e_po3030k_config_cam`

**Parameters**

<i>mode</i>	The given sampling mode
-------------	-------------------------

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 28 and [e\\_po3030k\\_config\\_cam](#)

**7.37.3.26 void e\_po3030k\_set\_sepia ( int status )**

Enable/Disable Sepia color

**Parameters**

<i>status</i>	Set <i>status</i> to 1 to enable, 0 to disable
---------------	--

**See also**

Datasheet p.34 and 74

7.37.3.27 void e\_po3030k\_set\_sepia\_tone ( unsigned char *cb*, unsigned char *cr* )

Set The color tone at sepia color condition

**Parameters**

<i>cb</i>	Cb tone
<i>cr</i>	Cr tone

**See also**

[e\\_po3030k\\_set\\_sepia](#) and Datasheet p. 41 and 74

7.37.3.28 int e\_po3030k\_set\_speed ( int *mode* )

Set the camera speed

**Warning**

This is an internal function, use e\_po3030k\_config\_cam

**Parameters**

<i>mode</i>	The given speed
-------------	-----------------

**Returns**

Zero if OK, non-zero if unknow mode

**See also**

Datasheet p. 26 and [e\\_po3030k\\_config\\_cam](#)

7.37.3.29 int e\_po3030k\_set\_vsync ( unsigned int *start*, unsigned int *stop*, unsigned int *col* )

Set the camera window VSYNC coordinate

**Warning**

This is an internal function, use e\_po3030k\_ConfigCam

**Parameters**

<i>start</i>	The start row
<i>stop</i>	The stop row
<i>col</i>	The start/stop column

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p.42-43, [e\\_po3030k\\_write\\_cam\\_registers](#) and [e\\_po3030k\\_config\\_cam](#)

7.37.3.30 int e\_po3030k\_set\_weight\_win ( unsigned int *x1*, unsigned int *x2*, unsigned int *y1*, unsigned int *y2* )

Set the Weighting Window coordinate

**Parameters**

<i>x1</i>	The X1 coordinate $\in \{211, x2\}$
<i>x2</i>	The X2 coordinate $\in \{x1 + 1, 423\}$
<i>y1</i>	The Y1 coordinate $\in \{160, y2\}$
<i>y2</i>	The Y2 coordinate $\in \{y1 + 1, 319\}$

**Returns**

Zero if OK, non-zero if an error occur

**See also**

Datasheet p. 49

7.37.3.31 void e\_po3030k\_set\_ww ( unsigned char *ww* )

Set the Center weight (Back Light compensation) Control parameter

**Parameters**

<i>ww</i>	Center weight ( 4 lower bits only )
-----------	-------------------------------------

**See also**

Datasheet p.44



7.37.3.32 int e\_po3030k\_set\_wx ( unsigned int *start*, unsigned int *stop* )

Set the camera window X coordinate

#### Warning

This is an internal function, use e\_po3030k\_ConfigCam

#### Parameters

<i>start</i>	The start column
<i>stop</i>	The stop column

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p.20-21, [e\\_po3030k\\_write\\_cam\\_registers](#), [e\\_po3030k\\_set\\_wy](#) and [e\\_po3030k\\_config\\_cam](#)

7.37.3.33 int e\_po3030k\_set\_wy ( unsigned int *start*, unsigned int *stop* )

Set the camera window Y coordinate

#### Warning

This is an internal function, use e\_po3030k\_ConfigCam

#### Parameters

<i>start</i>	The start row
<i>stop</i>	The stop row

#### Returns

Zero if OK, non-zero if an error occur

#### See also

Datasheet p.20-21, [e\\_po3030k\\_WriteCamRegisters](#), [e\\_po3030k\\_SetWX](#) and [e\\_po3030k\\_ConfigCam](#)

7.37.3.34 void e\_po3030k\_SetColorMatrix ( unsigned char *array*[3 \*3] )

Set color correction coefficient

**Parameters**

<i>array</i>	Color coefficient matrix (3x3), sign[7]   integer [6:5]   fractional [4:0]
--------------	--

**See also**

Datasheet p. 39

7.37.3.35 `int e_po3030k_sync_register_array ( unsigned char start, unsigned char stop )`

Write every known register between address *start* and *stop* (inclusively).

**Warning**

It's better to set the configuration with appropriate functions and then write all registers with `e_po3030k_↔ WriteCamRegisters`

**Parameters**

<i>start</i>	The beginning address of the write
<i>stop</i>	The last write address

**Returns**

The number of register written

**See also**

[e\\_po3030k\\_write\\_cam\\_registers](#)

7.37.3.36 `void e_po3030k_write_cam_registers ( void )`

The Po3030k module keep in memory the state of each register the camera has. When you configure the camera, it only alter the internal register state, not the camera one. This function write the internal register state in the camera.

**See also**

[e\\_po3030k\\_read\\_cam\\_registers](#)

7.37.3.37 `void e_po3030k_write_gamma_coef ( void )`

This special function write directly the Gamma coefficient and Gamma color select into camera register.

**Warning**

This function need extra care from the user

**See also**

[e\\_po3030k\\_set\\_gamma\\_coef](#)

7.37.3.38 `static long po3030k_get_pixelclock ( void ) [static]`

## 7.37.4 Variable Documentation

7.37.4.1 `unsigned char cam_reg[] [static]`

## 7.38 codec/e\_common.inc File Reference

## 7.39 codec/e\_sound.c File Reference

Package to play basics sounds on the e-puck's speaker.  
For more info look at this: [Sound](#).

```
#include "../motor_led/e_epuck_ports.h"  
#include "e_sound.h"
```

### Functions

- void [e\\_init\\_sound](#) (void)  
*Initialize all you need to play sound on speaker.*
- void [e\\_play\\_sound](#) (int sound\_nbr, int sound\_length)  
*Play a sound.*
- void [e\\_close\\_sound](#) (void)  
*Disable the sound After that you can't play sound anymore, if you want to, you have to call e\_init\_sound.*

### 7.39.1 Detailed Description

Package to play basics sounds on the e-puck's speaker.  
For more info look at this: [Sound](#).

#### Author

Code: Michael Bonani  
Doc: Jonathan Besuchet

### 7.39.2 Function Documentation

7.39.2.1 `void e_close_sound ( void )`

Disable the sound After that you can't play sound anymore, if you want to, you have to call e\_init\_sound.

7.39.2.2 `void e_init_sound ( void )`

Initialize all you need to play sound on speaker.

#### Warning

You must to call this function before playing a sound (call it only one time).

7.39.2.3 `void e_play_sound ( int sound_nbr, int sound_length )`

Play a sound.

**Parameters**

<code>sound_nbr</code>	the beginning of the sound
<code>sound_length</code>	the length of the sound

**See also**[Sound](#)

## 7.40 codec/e\_sound.h File Reference

Package to play basics sounds on the e-puck's speaker.

**Functions**

- void [e\\_init\\_sound](#) (void)  
*Initialize all you need to play sound on speaker.*
- void [e\\_play\\_sound](#) (int sound\_offset, int sound\_length)  
*Play a sound.*
- void [e\\_close\\_sound](#) (void)  
*Disable the sound After that you can't play sound anymore, if you want to, you have to call e\_init\_sound.*
- void [e\\_init\\_dci\\_master](#) (void)
- void [e\\_init\\_codec\\_slave](#) (void)
- void [e\\_sub\\_dci\\_kickoff](#) (int, int)

### 7.40.1 Detailed Description

Package to play basics sounds on the e-puck's speaker.

**Author**

Code: Michael Bonani  
Doc: Jonathan Besuchet

### 7.40.2 Function Documentation

#### 7.40.2.1 void e\_close\_sound ( void )

Disable the sound After that you can't play sound anymore, if you want to, you have to call e\_init\_sound.

7.40.2.2 void e\_init\_codec\_slave ( void )

7.40.2.3 void e\_init\_dci\_master ( void )

7.40.2.4 void e\_init\_sound ( void )

Initialize all you need to play sound on speaker.

#### Warning

You must to call this function before playing a sound (call it only one time).

7.40.2.5 void e\_play\_sound ( int *sound\_nbr*, int *sound\_length* )

Play a sound.

## Parameters

<code>sound_nbr</code>	the beginning of the sound
<code>sound_length</code>	the length of the sound

## See also

[Sound](#)

7.40.2.6 `void e_sub_dci_kickoff ( int , int )`

## 7.41 contrib/LIS\_sensors\_turret/e\_devantech.c File Reference

```
#include "e_devantech.h"
#include "../I2C/e_I2C_master_module.h"
#include "../I2C/e_I2C_protocol.h"
```

### Functions

- void [e\\_init\\_devantech](#) (void)
- void [e\\_disable\\_devantech](#) (void)
- unsigned int [e\\_get\\_dist\\_devantech](#) (char device\_add)
- unsigned int [e\\_get\\_delay\\_devantech](#) (char device\_add)
- char [e\\_get\\_sr\\_devantech](#) (char device\_add)
- unsigned int [e\\_get\\_light\\_devantech](#) (char device\_add)
- void [e\\_set\\_gain\\_devantech](#) (char device\_add, char gain)
- void [e\\_set\\_range\\_devantech](#) (char device\_add, char range)
- void [e\\_i2cd\\_write](#) (char device\_add, char reg, char value)
- char [e\\_i2cd\\_readb](#) (char device\_add, char reg)

#### 7.41.1 Function Documentation

7.41.1.1 `void e_disable_devantech ( void )`

7.41.1.2 `unsigned int e_get_delay_devantech ( char device_add )`

7.41.1.3 `unsigned int e_get_dist_devantech ( char device_add )`

7.41.1.4 `unsigned int e_get_light_devantech ( char device_add )`

7.41.1.5 `char e_get_sr_devantech ( char device_add )`

7.41.1.6 `char e_i2cd_readb ( char device_add, char reg )`

7.41.1.7 void `e_i2cd_write` ( char *device\_add*, char *reg*, char *value* )

7.41.1.8 void `e_init_devantech` ( void )

7.41.1.9 void `e_set_gain_devantech` ( char *device\_add*, char *gain* )

7.41.1.10 void `e_set_range_devantech` ( char *device\_add*, char *range* )

## 7.42 contrib/LIS\_sensors\_turret/e\_devantech.h File Reference

Devantech sensor of e-puck.

```
#include "../I2C/e_I2C_master_module.h"
#include "../motor_led/e_epuck_ports.h"
```

### Functions

- void `e_init_devantech` (void)
- void `e_disable_devantech` (void)
- unsigned int `e_get_dist_devantech` (char *device\_add*)
- unsigned int `e_get_delay_devantech` (char *device\_add*)
- char `e_get_sr_devantech` (char *device\_add*)
- unsigned int `e_get_light_devantech` (char *device\_add*)
- void `e_set_gain_devantech` (char *device\_add*, char *gain*)
- void `e_set_range_devantech` (char *device\_add*, char *range*)
- void `e_i2cd_write` (char *device\_add*, char *reg*, char *value*)
- char `e_i2cd_readb` (char *device\_add*, char *reg*)
- unsigned int `e_i2cd_readw` (char *device\_add*, char *reg*)

### 7.42.1 Detailed Description

Devantech sensor of e-puck.

#### Author

Jonathan Besuchet

## 7.42.2 Function Documentation

7.42.2.1 void `e_disable_devantech` ( void )

7.42.2.2 unsigned int `e_get_delay_devantech` ( char *device\_add* )

7.42.2.3 unsigned int `e_get_dist_devantech` ( char *device\_add* )

7.42.2.4 unsigned int `e_get_light_devantech` ( char *device\_add* )

7.42.2.5 char `e_get_sr_devantech` ( char *device\_add* )

7.42.2.6 char `e_i2cd_readb` ( char *device\_add*, char *reg* )

7.42.2.7 unsigned int `e_i2cd_readw` ( char *device\_add*, char *reg* )

7.42.2.8 void `e_i2cd_write` ( char *device\_add*, char *reg*, char *value* )

7.42.2.9 void `e_init_devantech` ( void )

7.42.2.10 void `e_set_gain_devantech` ( char *device\_add*, char *gain* )

7.42.2.11 void `e_set_range_devantech` ( char *device\_add*, char *range* )

## 7.43 contrib/LIS\_sensors\_turret/e\_sensex.c File Reference

```
#include "e_sensex.h"  
#include "e_sharp.h"  
#include <a_d/advance_ad_scan/e_acc.h>  
#include <motor_led/advance_one_timer/e_agenda.h>  
#include <I2C/e_I2C_master_module.h>  
#include <I2C/e_I2C_protocol.h>
```

### Functions

- void `e_stop_sensex_wait` (void)
- void `e_start_sensex_wait` (void)
- int `e_get_sensex_wait` (void)
- void `e_init_sensex` (void)
- int `e_sensex_process` (int \**sensex\_param*, unsigned int \**sensex\_value*)

### Variables

- static int `sensex_wait` =0



### 7.43.1 Function Documentation

7.43.1.1 `int e_get_sensex_wait ( void )`

7.43.1.2 `void e_init_sensex ( void )`

7.43.1.3 `int e_sensex_process ( int * sensex_param, unsigned int * sensex_value )`

!!WALTER!!!!

!!WALTER!!!!

7.43.1.4 `void e_start_sensex_wait ( void )`

7.43.1.5 `void e_stop_sensex_wait ( void )`

### 7.43.2 Variable Documentation

7.43.2.1 `int sensext_wait =0 [static]`

## 7.44 contrib/LIS\_sensors\_turret/e\_sensex.h File Reference

### Macros

- `#define I2C_ADDR_SENSEXT 0b10100010`
- `#define I2C_ADDR_SRF08 0xE0`
- `#define I2C_ADDR_SRF10 0xE0`
- `#define I2C_ADDR_CMPS03 0xC0`
- `#define I2C_ADDR_SRF235 0xE0`

### Functions

- `void e_init_sensex (void)`
- `int e_sensex_process (int *sensex_param, unsigned int *sensex_value)`
- `void e_stop_sensex_wait (void)`
- `void e_start_sensex_wait (void)`
- `int e_get_sensex_wait (void)`

## 7.44.1 Macro Definition Documentation

7.44.1.1 `#define I2C_ADDR_CMPS03 0xC0`

7.44.1.2 `#define I2C_ADDR_SENSEXT 0b10100010`

7.44.1.3 `#define I2C_ADDR_SRF08 0xE0`

7.44.1.4 `#define I2C_ADDR_SRF10 0xE0`

7.44.1.5 `#define I2C_ADDR_SRF235 0xE0`

## 7.44.2 Function Documentation

7.44.2.1 `int e_get_sensex_wait ( void )`

7.44.2.2 `void e_init_sensex ( void )`

7.44.2.3 `int e_sensex_process ( int * sensex_param, unsigned int * sensex_value )`

!!WALTER!!!!

!!WALTER!!!!

7.44.2.4 `void e_start_sensex_wait ( void )`

7.44.2.5 `void e_stop_sensex_wait ( void )`

## 7.45 contrib/LIS\_sensors\_turret/e\_sharp.c File Reference

```
#include "a_d/e_ad_conv.h"
#include "motor_led/e_epuck_ports.h"
#include "e_sharp.h"
#include <a_d/advance_ad_scan/e_acc.h>
```

### Functions

- void [e\\_init\\_sharp](#) (void)
- int [e\\_get\\_dist\\_sharp](#) ()
- void [e\\_set\\_sharp\\_led](#) (unsigned int sharp\_led\_number, unsigned int value)
- void [e\\_sharp\\_led\\_clear](#) (void)
- void [e\\_sharp\\_on](#) (void)
- void [e\\_sharp\\_off](#) (void)

## 7.45.1 Function Documentation

7.45.1.1 int e\_get\_dist\_sharp ( void )

7.45.1.2 void e\_init\_sharp ( void )

7.45.1.3 void e\_set\_sharp\_led ( unsigned int *sharp\_led\_number*, unsigned int *value* )

7.45.1.4 void e\_sharp\_led\_clear ( void )

7.45.1.5 void e\_sharp\_off ( void )

7.45.1.6 void e\_sharp\_on ( void )

## 7.46 contrib/LIS\_sensors\_turret/e\_sharp.h File Reference

### Macros

- #define [SHARP](#) 5
- #define [SHARP\\_LED1\\_LATG6](#)
- #define [SHARP\\_LED2\\_LATG7](#)
- #define [SHARP\\_LED3\\_LATG8](#)
- #define [SHARP\\_LED4\\_LATG9](#)
- #define [SHARP\\_LED5\\_LATB6](#)
- #define [SHARP\\_VIN\\_LATB7](#)
- #define [SHARP\\_LED1\\_DIR\\_TRISG6](#)
- #define [SHARP\\_LED2\\_DIR\\_TRISG7](#)
- #define [SHARP\\_LED3\\_DIR\\_TRISG8](#)
- #define [SHARP\\_LED4\\_DIR\\_TRISG9](#)
- #define [SHARP\\_LED5\\_DIR\\_TRISB6](#)
- #define [SHARP\\_VIN\\_DIR\\_TRISB7](#)

### Functions

- void [e\\_init\\_sharp](#) (void)
- int [e\\_get\\_dist\\_sharp](#) (void)
- void [e\\_set\\_sharp\\_led](#) (unsigned int sharp\_led\_number, unsigned int value)
- void [e\\_sharp\\_led\\_clear](#) (void)
- void [e\\_sharp\\_on](#) (void)
- void [e\\_sharp\\_off](#) (void)

## 7.46.1 Macro Definition Documentation

7.46.1.1 `#define SHARP 5`

7.46.1.2 `#define SHARP_LED1_LATG6`

7.46.1.3 `#define SHARP_LED1_DIR_TRISG6`

7.46.1.4 `#define SHARP_LED2_LATG7`

7.46.1.5 `#define SHARP_LED2_DIR_TRISG7`

7.46.1.6 `#define SHARP_LED3_LATG8`

7.46.1.7 `#define SHARP_LED3_DIR_TRISG8`

7.46.1.8 `#define SHARP_LED4_LATG9`

7.46.1.9 `#define SHARP_LED4_DIR_TRISG9`

7.46.1.10 `#define SHARP_LED5_LATB6`

7.46.1.11 `#define SHARP_LED5_DIR_TRISB6`

7.46.1.12 `#define SHARP_VIN_LATB7`

7.46.1.13 `#define SHARP_VIN_DIR_TRISB7`

## 7.46.2 Function Documentation

7.46.2.1 `int e_get_dist_sharp ( void )`

7.46.2.2 `void e_init_sharp ( void )`

7.46.2.3 `void e_set_sharp_led ( unsigned int sharp_led_number, unsigned int value )`

7.46.2.4 `void e_sharp_led_clear ( void )`

7.46.2.5 `void e_sharp_off ( void )`

7.46.2.6 `void e_sharp_on ( void )`

## 7.47 contrib/SWIS\_com\_module/ComModule.c File Reference

```
#include "ComModule.h"
#include <stdlib.h>
#include "../motor_led/e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../I2C/e_I2C_master_module.h"
```

## Macros

- #define [COM\\_MODULE\\_I2C\\_ADDR](#) (0x3F<<1)
- #define [BUFFER\\_DATA\\_LENGTH](#) (COM\_MODULE\_MAXSIZE + 14)
- #define [STATUS\\_REG\\_ADDR](#) 0x00
- #define [CONFIG\\_REG\\_ADDR](#) 0x01
- #define [SEND\\_REG\\_ADDR](#) 0x02
- #define [SOFTATT\\_REG\\_ADDR](#) 0x03
- #define [OWNGROUP\\_REG\\_ADDR](#) 0x04
- #define [OWNADDRL\\_REG\\_ADDR](#) 0x05
- #define [OWNADDRH\\_REG\\_ADDR](#) 0x06
- #define [SEND\\_BUFFER\\_START](#) 0x07
- #define [SEND\\_BUFFER\\_END](#) (SEND\_BUFFER\_START + BUFFER\_DATA\_LENGTH)
- #define [REC\\_BUFFER\\_START](#) (SEND\_BUFFER\_END + 1)
- #define [REC\\_BUFFER\\_END](#) (REC\_BUFFER\_START + BUFFER\_DATA\_LENGTH)
- #define [AM\\_MSGTYPE](#) 0x0A
- #define [AM\\_MSGTYPE\\_IN\\_PACKET\\_OFFSET](#) 0x08
- #define [ADDRLDATA\\_IN\\_PACKET\\_OFFSET](#) 0x06
- #define [ADDRHDATA\\_IN\\_PACKET\\_OFFSET](#) 0x07
- #define [TYPEDATA\\_IN\\_PACKET\\_OFFSET](#) 0x08
- #define [GROUPDATA\\_IN\\_PACKET\\_OFFSET](#) 0x09
- #define [FIRSTDATA\\_IN\\_PACKET\\_OFFSET](#) 0x0A
- #define [PACKET\\_READY\\_FLAG](#) 0x01
- #define [TX\\_IDLE\\_FLAG](#) 0x02
- #define [PACKET\\_LOST\\_FLAG](#) 0x04
- #define [TX\\_SEND\\_ERROR](#) 0x08
- #define [HARDWAREATT\\_SET\\_FLAG](#) 0x01
- #define [RADIO\\_ENABLED\\_FLAG](#) 0x80
- #define [REQUEST\\_TO\\_SEND\\_FLAG](#) 0x01

## Functions

- unsigned char [ReadRegister](#) (unsigned char registeraddr)
- void [WriteRegister](#) (unsigned char registeraddr, unsigned char value)
- void [InitComModule](#) (unsigned char owngroup, unsigned int ownaddress, unsigned char hardwareattenuatormode, unsigned char softwareattenuatorvalue)
- int [IsModulePlugged](#) ()
- void [SetRadioEnabledState](#) (unsigned char mode)
- void [SetHardwareAttenuator](#) (unsigned char attenuatormode)
- void [SetSoftwareAttenuator](#) (unsigned char attenuatorvalue)
- void [SetOwnGroup](#) (unsigned char owngroup)
- void [SetOwnAddress](#) (unsigned int ownaddress)
- unsigned char [GetHardwareAttenuator](#) ()
- unsigned char [GetRadioEnabledState](#) ()
- unsigned char [GetSoftwareAttenuator](#) ()
- unsigned char [GetOwnGroup](#) ()
- unsigned int [GetOwnAddress](#) ()
- unsigned char [GetStatus](#) ()
- int [SendPacket](#) (unsigned char destinationgroup, unsigned int destinationaddress, unsigned char \*packet, int packetsize)
- int [IsPacketReady](#) (unsigned char \*packet, int \*packetsize)

## 7.47.1 Macro Definition Documentation

7.47.1.1 #define ADDRHDATA\_IN\_PACKET\_OFFSET 0x07

7.47.1.2 #define ADDRDATA\_IN\_PACKET\_OFFSET 0x06

7.47.1.3 #define AM\_MSGTYPE 0x0A

7.47.1.4 #define AM\_MSGTYPE\_IN\_PACKET\_OFFSET 0x08

7.47.1.5 #define BUFFER\_DATA\_LENGTH (COM\_MODULE\_MAXSIZE + 14)

7.47.1.6 #define COM\_MODULE\_I2C\_ADDR (0x3F<<1)

7.47.1.7 #define CONFIG\_REG\_ADDR 0x01

7.47.1.8 #define FIRSTDATA\_IN\_PACKET\_OFFSET 0x0A

7.47.1.9 #define GROUDDATA\_IN\_PACKET\_OFFSET 0x09

7.47.1.10 #define HARDWAREATT\_SET\_FLAG 0x01

7.47.1.11 #define OWNADDRH\_REG\_ADDR 0x06

7.47.1.12 #define OWNADDRL\_REG\_ADDR 0x05

7.47.1.13 #define OWNGROUP\_REG\_ADDR 0x04

7.47.1.14 #define PACKET\_LOST\_FLAG 0x04

7.47.1.15 #define PACKET\_READY\_FLAG 0x01

7.47.1.16 #define RADIO\_ENABLED\_FLAG 0x80

7.47.1.17 #define REC\_BUFFER\_END (REC\_BUFFER\_START + BUFFER\_DATA\_LENGTH)

7.47.1.18 #define REC\_BUFFER\_START (SEND\_BUFFER\_END + 1)

7.47.1.19 #define REQUEST\_TO\_SEND\_FLAG 0x01

7.47.1.20 #define SEND\_BUFFER\_END (SEND\_BUFFER\_START + BUFFER\_DATA\_LENGTH)

7.47.1.21 #define SEND\_BUFFER\_START 0x07

7.47.1.22 #define SEND\_REG\_ADDR 0x02

- 7.47.1.23 `#define SOFTATT_REG_ADDR 0x03`
- 7.47.1.24 `#define STATUS_REG_ADDR 0x00`
- 7.47.1.25 `#define TX_IDLE_FLAG 0x02`
- 7.47.1.26 `#define TX_SEND_ERROR 0x08`
- 7.47.1.27 `#define TYPEDATA_IN_PACKET_OFFSET 0x08`

## 7.47.2 Function Documentation

- 7.47.2.1 `unsigned char GetHardwareAttenuator ( )`
- 7.47.2.2 `unsigned int GetOwnAddress ( )`
- 7.47.2.3 `unsigned char GetOwnGroup ( )`
- 7.47.2.4 `unsigned char GetRadioEnabledState ( )`
- 7.47.2.5 `unsigned char GetSoftwareAttenuator ( )`
- 7.47.2.6 `unsigned char GetStatus ( )`
- 7.47.2.7 `void InitComModule ( unsigned char owngroup, unsigned int ownaddress, unsigned char hardwareattenuatormode, unsigned char softwareattenuatorvalue )`
- 7.47.2.8 `int IsModulePlugged ( )`
- 7.47.2.9 `int IsPacketReady ( unsigned char * packet, int * packetsize )`
- 7.47.2.10 `unsigned char ReadRegister ( unsigned char registeraddr )`
- 7.47.2.11 `int SendPacket ( unsigned char destinationgroup, unsigned int destinationaddress, unsigned char * packet, int packetsize )`
- 7.47.2.12 `void SetHardwareAttenuator ( unsigned char attenuatormode )`
- 7.47.2.13 `void SetOwnAddress ( unsigned int ownaddress )`
- 7.47.2.14 `void SetOwnGroup ( unsigned char owngroup )`
- 7.47.2.15 `void SetRadioEnabledState ( unsigned char mode )`
- 7.47.2.16 `void SetSoftwareAttenuator ( unsigned char attenuatorvalue )`
- 7.47.2.17 `void WriteRegister ( unsigned char registeraddr, unsigned char value )`

## 7.48 contrib/SWIS\_com\_module/ComModule.h File Reference

Radio communication.

```
#include "p30f6014A.h"
```

## Macros

- `#define COM_MODULE_HW_ATTENUATOR_25DB 1`
- `#define COM_MODULE_HW_ATTENUATOR_0DB 0`
- `#define COM_MODULE_DEFAULT_GROUP 0x7d`
- `#define COM_MODULE_MAXSIZE 108`

## Functions

- void `InitComModule` (unsigned char owngroup, unsigned int ownaddress, unsigned char hardwareattenuator-mode, unsigned char softwareattenuatorvalue)
- int `IsModulePlugged` ()
- void `SetRadioEnabledState` (unsigned char mode)
- void `SetHardwareAttenuator` (unsigned char AttenuatorMode)
- void `SetSoftwareAttenuator` (unsigned char AttenuatorValue)
- void `SetOwnGroup` (unsigned char GroupID)
- void `SetOwnAddress` (unsigned int ownaddress)
- unsigned char `GetRadioEnabledState` ()
- unsigned char `GetHardwareAttenuator` ()
- unsigned char `GetSoftwareAttenuator` ()
- unsigned char `GetOwnGroup` ()
- unsigned char `GetStatus` ()
- int `SendPacket` (unsigned char destinationgroup, unsigned int destinationaddress, unsigned char \*packet, int packetSize)
- int `IsPacketReady` (unsigned char \*packet, int \*packetSize)

### 7.48.1 Detailed Description

Radio communication.

#### Author

Jonathan Besuchet

### 7.48.2 Macro Definition Documentation

7.48.2.1 `#define COM_MODULE_DEFAULT_GROUP 0x7d`

7.48.2.2 `#define COM_MODULE_HW_ATTENUATOR_0DB 0`

7.48.2.3 `#define COM_MODULE_HW_ATTENUATOR_25DB 1`

7.48.2.4 `#define COM_MODULE_MAXSIZE 108`

### 7.48.3 Function Documentation

7.48.3.1 unsigned char `GetHardwareAttenuator` ( )



- 7.48.3.2 unsigned char GetOwnGroup ( )
- 7.48.3.3 unsigned char GetRadioEnabledState ( )
- 7.48.3.4 unsigned char GetSoftwareAttenuator ( )
- 7.48.3.5 unsigned char GetStatus ( )
- 7.48.3.6 void InitComModule ( unsigned char *owngroup*, unsigned int *ownaddress*, unsigned char *hardwareattenuatormode*, unsigned char *softwareattenuatorvalue* )
- 7.48.3.7 int IsModulePlugged ( )
- 7.48.3.8 int IsPacketReady ( unsigned char \* *packet*, int \* *packetSize* )
- 7.48.3.9 int SendPacket ( unsigned char *destinationgroup*, unsigned int *destinationaddress*, unsigned char \* *packet*, int *packetsize* )
- 7.48.3.10 void SetHardwareAttenuator ( unsigned char *AttenuatorMode* )
- 7.48.3.11 void SetOwnAddress ( unsigned int *ownaddress* )
- 7.48.3.12 void SetOwnGroup ( unsigned char *GroupID* )
- 7.48.3.13 void SetRadioEnabledState ( unsigned char *mode* )
- 7.48.3.14 void SetSoftwareAttenuator ( unsigned char *AttenuatorValue* )

## 7.49 fft/e\_fft.c File Reference

Package to manage the FFT.

```
#include <dsp.h>
#include "e_fft.h"
```

### Functions

- const fractcomplex twiddleFactors[FFT\_BLOCK\_LENGTH/2] `__attribute__` ((space(auto\_psv), aligned(FFT\_BLOCK\_LENGTH \*2)))
- void `e_doFFT_asm` (fractcomplex \*sigCmpx)  
*Execute the FFT with dsPic special instructions.*

### 7.49.1 Detailed Description

Package to manage the FFT.

#### Author

Code & doc: Jonathan Besuchet

### 7.49.2 Function Documentation

7.49.2.1 `const fractcomplex twiddleFactors [FFT_BLOCK_LENGTH/2] __attribute__ ( (space(auto_psv), aligned(FFT_BLOCK_LENGTH *2)) )`

7.49.2.2 `void e_doFFT_asm ( fractcomplex * sigCmpx )`

Execute the FFT with dsPic special instructions.

#### Parameters

<code>sigCmpx</code>	The pointer of the beginning of the array on which you want to perform the FFT.
----------------------	---

## 7.50 fft/e\_fft.h File Reference

Package to manage the FFT.

```
#include <dsp.h>
```

### Macros

- `#define FFT_BLOCK_LENGTH 256`
- `#define LOG2_BLOCK_LENGTH 8`

### Functions

- void `e_doFFT_asm` (fractcomplex \*sigCmpx)  
*Execute the FFT with dsPic special instructions.*

### 7.50.1 Detailed Description

Package to manage the FFT.

In this package the FFT (fast fourier transform) is done with the special dsPic 30fxxx instructions.

Before calling the `e_doFFT_asm()` function, you must to fill the sigCmpx array with your values. This array is stocked in the Y memory of the dsPic.

#### Author

Code & doc: Jonathan Besuchet

## 7.50.2 Macro Definition Documentation

7.50.2.1 `#define FFT_BLOCK_LENGTH 256`

7.50.2.2 `#define LOG2_BLOCK_LENGTH 8`

## 7.50.3 Function Documentation

7.50.3.1 `void e_doFFT_asm ( fractcomplex * sigCmpx )`

Execute the FFT with dsPic special instructions.

### Parameters

<code>sigCmpx</code>	The pointer of the beginning of the array on which you want to perform the FFT.
----------------------	---

## 7.51 `fft/e_fft_utilities.h` File Reference

Some fft features.

### Functions

- void `e_fast_copy` (int \*in\_array, int \*out\_array, int size)  
*Copy an array to an other array, using REPEAT instruction.*
- void `e_subtract_mean` (int \*array, int size, int log2size)  
*Subtract the mean from the samples of the array (-> produce zero-mean samples)*

### 7.51.1 Detailed Description

Some fft features.

Here you have two functions that are really usefull to work with the FFT

#### Author

Code & doc: Jonathan Besuchet

### 7.51.2 Function Documentation

7.51.2.1 `void e_fast_copy ( int * in_array, int * out_array, int size )`

Copy an array to an other array, using REPEAT instruction.

## Parameters

<i>in_array</i>	The array from which you want to copy
<i>out_array</i>	The destination array
<i>size</i>	The number of element you want to copy

## 7.51.2.2 void e\_subtract\_mean ( int \* array, int size, int log2size )

Subtract the mean from the samples of the array (-> produce zero-mean samples)

## Parameters

<i>array</i>	The array that you want to produce zero-mean samples
<i>size</i>	The size of the array
<i>log2size</i>	The log in base 2 of the size

## 7.52 fft/e\_input\_signal.c File Reference

Allocate memory and initialize the sigCmpx array.

```
#include <dsp.h>
#include "e_fft.h"
```

## Functions

- `fractcomplex sigCmpx[FFT_BLOCK_LENGTH] __attribute__((section(".ydata, data, ymemory"), aligned(FFT_BLOCK_LENGTH * 2 * 2)))`

## 7.52.1 Detailed Description

Allocate memory and initialize the sigCmpx array.

sigCmpx is the main array in which the FFT will be done. So you must to fill this array with your values to perform the FFT.

This array has the following size: 64, 128, 256, 512 depending the choice you have made in fft.h

## Author

Code & doc: Jonathan Besuchet

## 7.52.2 Function Documentation

- 7.52.2.1 `fractcomplex sigCmpx [FFT_BLOCK_LENGTH] __attribute__ ( (section(".ydata, data, ymemory"), aligned(FFT_BLOCK_LENGTH *2 *2)) )`

## 7.53 fft/e\_twiddle\_factors.c File Reference

The FFT factor from Microchip.

### Functions

- `const fractcomplex twiddleFactors[] __attribute__ ((space(auto_psv), aligned(FFT_BLOCK_LENGTH *2)))`

### 7.53.1 Detailed Description

The FFT factor from Microchip.

#### Author

Jonathan Besuchet

## 7.53.2 Function Documentation

- 7.53.2.1 `const fractcomplex twiddleFactors [] __attribute__ ( (space(auto_psv), aligned(FFT_BLOCK_LENGTH *2)) )`

## 7.54 I2C/e\_I2C\_master\_module.c File Reference

Manage I2C basics.

```
#include "e_I2C_master_module.h"
```

### Functions

- void `idle_i2c` (void)  
*Wait until I2C Bus is Inactive.*
- char `e_i2c_init` (void)  
*Initialize the microcontroller for I2C uses.*
- char `e_i2c_deinit` (void)
- char `e_i2c_reset` (void)  
*Reset the microcontroller for I2C uses.*
- char `e_i2c_enable` (void)  
*Enable special I2C interrupt.*
- char `e_i2c_disable` (void)  
*Disable special I2C interrupt.*
- char `e_i2c_start` (void)

- *Make the start bit.*  
char `e_i2c_restart` (void)
- *Make the restart bit.*  
char `e_i2c_stop` (void)
- *Make the stop bit.*  
char `e_i2c_ack` (void)
- *Make the acknowledgement bit.*  
char `e_i2c_nack` (void)
- *Make the non-acknowledgement bit.*  
char `e_i2c_read` (char \*buf)
- *Read the I2C input register.*  
char `e_i2c_write` (char byte)
- *Write on the I2C output register.*  
void `__attribute__` ((\_\_interrupt\_\_, auto\_psv))

## Variables

- char `e_i2c_mode`
- int `e_interrupts` [3]

### 7.54.1 Detailed Description

Manage I2C basics.

This module manage the I2C basics functions (low level I2C functions).  
They are made to perform the basics tasks like:

- initializing the I2C on the microcontroller
- sending the Start bit (`e_i2c_start`)
- sending the Restart bit (`e_i2c_restart`)
- sending the Stop bit (`e_i2c_stop`)
- sending the acknowledgement bit (`e_i2c_ack`)
- writing or receiving a byte (`e_i2c_write`, `e_i2c_read`)
- ...

#### Author

Code: Davis Daidie  
Doc: Jonathan Besuchet

### 7.54.2 Function Documentation

7.54.2.1 void `__attribute__` ( (\_\_interrupt\_\_, auto\_psv) )

7.54.2.2 char `e_i2c_ack` ( void )

Make the acknowledgement bit.

#### Returns

1 to confirme the operation and 0 for an error

7.54.2.3 `char e_i2c_deinit ( void )`

7.54.2.4 `char e_i2c_disable ( void )`

Disable special I2C interrupt.

**Returns**

1 to confirm the operation and 0 for an error

7.54.2.5 `char e_i2c_enable ( void )`

Enable special I2C interrupt.

**Returns**

1 to confirm the operation and 0 for an error

7.54.2.6 `char e_i2c_init ( void )`

Initialize the microcontroller for I2C uses.

**Returns**

1 to confirm the operation and 0 for an error

7.54.2.7 `char e_i2c_nack ( void )`

Make the non-acknowledgement bit.

**Returns**

1 to confirm the operation and 0 for an error

7.54.2.8 `char e_i2c_read ( char * buf )`

Read the I2C input register.

**Parameters**

<i>buf</i>	A pointer to store the data received
------------	--------------------------------------

**Returns**

1 to confirm the operation and 0 for an error

**7.54.2.9** `char e_i2c_reset ( void )`

Reset the microcontroller for I2C uses.

**Returns**

1 to confirm the operation and 0 for an error

**7.54.2.10** `char e_i2c_restart ( void )`

Make the restart bit.

**Returns**

1 to confirm the operation and 0 for an error

**7.54.2.11** `char e_i2c_start ( void )`

Make the start bit.

**Returns**

1 to confirm the operation and 0 for an error

**7.54.2.12** `char e_i2c_stop ( void )`

Make the stop bit.

**Returns**

1 to confirm the operation and 0 for an error

**7.54.2.13** `char e_i2c_write ( char byte )`

Write on the I2C output register.

**Parameters**

<i>byte</i>	What you want to send on I2C
-------------	------------------------------



## Returns

1 to confirm the operation and 0 for an error

7.54.2.14 void idle\_i2c ( void )

Wait until I2C Bus is Inactive.

## 7.54.3 Variable Documentation

7.54.3.1 char e\_i2c\_mode

7.54.3.2 int e\_interrupts[3]

## 7.55 I2C/e\_I2C\_master\_module.h File Reference

Manage I2C basics.

```
#include "p30F6014A.h"
```

## Macros

- #define [START](#) 1
- #define [WRITE](#) 2
- #define [ACKNOWLEDGE](#) 3
- #define [READ](#) 4
- #define [STOP](#) 5
- #define [RESTART](#) 6
- #define [ERROR](#) 10
- #define [OPERATION\\_OK](#) 0

## Functions

- char [e\\_i2c\\_init](#) (void)  
*Initialize the microcontroller for I2C uses.*
- char [e\\_i2c\\_deinit](#) (void)
- char [e\\_i2c\\_start](#) (void)  
*Make the start bit.*
- char [e\\_i2c\\_restart](#) (void)  
*Make the restart bit.*
- char [e\\_i2c\\_ack](#) (void)  
*Make the acknowledgement bit.*
- char [e\\_i2c\\_nack](#) (void)  
*Make the non-acknowledgement bit.*
- char [e\\_i2c\\_read](#) (char \*buf)  
*Read the I2C input register.*

- char `e_i2c_stop` (void)  
*Make the stop bit.*
- char `e_i2c_write` (char byte)  
*Write on the I2C output register.*
- char `e_i2c_enable` (void)  
*Enable special I2C interrupt.*
- char `e_i2c_disable` (void)  
*Disable special I2C interrupt.*
- char `e_i2c_reset` (void)  
*Reset the microcontroller for I2C uses.*

### 7.55.1 Detailed Description

Manage I2C basics.

#### Author

Code: Davis Daidie

Doc: Jonathan Besuchet

### 7.55.2 Macro Definition Documentation

7.55.2.1 `#define ACKNOWLEDGE 3`

7.55.2.2 `#define ERROR 10`

7.55.2.3 `#define OPERATION_OK 0`

7.55.2.4 `#define READ 4`

7.55.2.5 `#define RESTART 6`

7.55.2.6 `#define START 1`

7.55.2.7 `#define STOP 5`

7.55.2.8 `#define WRITE 2`

### 7.55.3 Function Documentation

7.55.3.1 `char e_i2c_ack ( void )`

Make the acknowledgement bit.

#### Returns

1 to confirme the operation and 0 for an error

7.55.3.2 `char e_i2c_deinit ( void )`

7.55.3.3 `char e_i2c_disable ( void )`

Disable special I2C interrupt.

**Returns**

1 to confirm the operation and 0 for an error

7.55.3.4 `char e_i2c_enable ( void )`

Enable special I2C interrupt.

**Returns**

1 to confirm the operation and 0 for an error

7.55.3.5 `char e_i2c_init ( void )`

Initialize the microcontroller for I2C uses.

**Returns**

1 to confirm the operation and 0 for an error

7.55.3.6 `char e_i2c_nack ( void )`

Make the non-acknowledgement bit.

**Returns**

1 to confirm the operation and 0 for an error

7.55.3.7 `char e_i2c_read ( char * buf )`

Read the I2C input register.

**Parameters**

<i>buf</i>	A pointer to store the data received
------------	--------------------------------------

**Returns**

1 to confirm the operation and 0 for an error

**7.55.3.8** `char e_i2c_reset ( void )`

Reset the microcontroller for I2C uses.

**Returns**

1 to confirm the operation and 0 for an error

**7.55.3.9** `char e_i2c_restart ( void )`

Make the restart bit.

**Returns**

1 to confirm the operation and 0 for an error

**7.55.3.10** `char e_i2c_start ( void )`

Make the start bit.

**Returns**

1 to confirm the operation and 0 for an error

**7.55.3.11** `char e_i2c_stop ( void )`

Make the stop bit.

**Returns**

1 to confirm the operation and 0 for an error

**7.55.3.12** `char e_i2c_write ( char byte )`

Write on the I2C output register.

**Parameters**

<i>byte</i>	What you want to send on I2C
-------------	------------------------------

**Returns**

1 to confirm the operation and 0 for an error

## 7.56 I2C/e\_I2C\_protocol.c File Reference

Manage I2C protocole.

```
#include "e_I2C_protocol.h"
```

**Functions**

- void [e\\_i2cp\\_init](#) (void)  
*Initialize the microcontroller for I2C uses.*
- void [e\\_i2cp\\_deinit](#) (void)
- void [e\\_i2cp\\_enable](#) (void)  
*Enable special I2C interrupt.*
- void [e\\_i2cp\\_disable](#) (void)  
*Disable special I2C interrupt.*
- char [e\\_i2cp\\_read](#) (char device\_add, char reg)  
*Read a specific register on a device.*
- char [e\\_i2cp\\_write](#) (char device\_add, char reg, char value)  
*Write a specific register on a device.*

### 7.56.1 Detailed Description

Manage I2C protocole.

This module manage the I2C protocole. The function's module are made to directly send or receive data from or to a specified slave.

**Warning**

This file must be include to communicate with an PO3030K camera throught the I2C communication protocol

**Author**

Code: Davis Daidie  
Doc: Jonathan Besuchet

### 7.56.2 Function Documentation

7.56.2.1 void [e\\_i2cp\\_deinit](#) ( void )

7.56.2.2 void [e\\_i2cp\\_disable](#) ( void )

Disable special I2C interrupt.

**Returns**

1 to confirm the operation and 0 for an error

### 7.56.2.3 void e\_i2cp\_enable ( void )

Enable special I2C interrupt.

#### Returns

1 to confirm the operation and 0 for an error

### 7.56.2.4 void e\_i2cp\_init ( void )

Initialize the microcontroller for I2C uses.

#### Returns

1 to confirm the operation and 0 for an error

### 7.56.2.5 char e\_i2cp\_read ( char *device\_add*, char *reg* )

Read a specific register on a device.

#### Parameters

<i>device_add</i>	The address of the device you want information
<i>reg</i>	The register address you want read on the device

#### Returns

The readed value

### 7.56.2.6 char e\_i2cp\_write ( char *device\_add*, char *reg*, char *value* )

Write a specific register on a device.

#### Parameters

<i>device_add</i>	The address of the device you want information
<i>reg</i>	The register address you want read on the device
<i>value</i>	The data you want to write

#### Returns

1 to confirm the operation and 0 for an error

## 7.57 I2C/e\_I2C\_protocol.h File Reference

Manage I2C protocole.

```
#include "e_I2C_master_module.h"
#include "../motor_led/e_epuck_ports.h"
```

### Functions

- void `e_i2cp_init` (void)  
*Initialize the microcontroller for I2C uses.*
- void `e_i2cp_deinit` (void)
- char `e_i2cp_write` (char device\_add, char reg, char value)  
*Write a specific register on a device.*
- char `e_i2cp_read` (char device\_add, char reg)  
*Read a specific register on a device.*
- char `e_i2cp_read_string` (char device\_add, unsigned char read\_buffer[], char start\_address, char string\_↔ length)
- char `e_i2cp_write_string` (char device\_add, unsigned char write\_buffer[], char start\_address, char string\_↔ length)
- void `e_i2cp_enable` (void)  
*Enable special I2C interrupt.*
- void `e_i2cp_disable` (void)  
*Disable special I2C interrupt.*

### 7.57.1 Detailed Description

Manage I2C protocole.

This module manage the I2C protocole. The function's module are made to directly send or receive data from or to a specified slave.

#### Warning

This file must be include to communicate with an PO3030K camera throught the I2C communication protocol

#### Author

Code: Davis Daidie  
Doc: Jonathan Besuchet

### 7.57.2 Function Documentation

7.57.2.1 void `e_i2cp_deinit` ( void )

7.57.2.2 void `e_i2cp_disable` ( void )

Disable special I2C interrupt.

#### Returns

1 to confirme the operation and 0 for an error

### 7.57.2.3 void e\_i2cp\_enable ( void )

Enable special I2C interrupt.

#### Returns

1 to confirm the operation and 0 for an error

### 7.57.2.4 void e\_i2cp\_init ( void )

Initialize the microcontroller for I2C uses.

#### Returns

1 to confirm the operation and 0 for an error

### 7.57.2.5 char e\_i2cp\_read ( char *device\_add*, char *reg* )

Read a specific register on a device.

#### Parameters

<i>device_add</i>	The address of the device you want information
<i>reg</i>	The register address you want read on the device

#### Returns

The readed value

### 7.57.2.6 char e\_i2cp\_read\_string ( char *device\_add*, unsigned char *read\_buffer*[], char *start\_address*, char *string\_length* )

### 7.57.2.7 char e\_i2cp\_write ( char *device\_add*, char *reg*, char *value* )

Write a specific register on a device.

#### Parameters

<i>device_add</i>	The address of the device you want information
<i>reg</i>	The register address you want read on the device
<i>value</i>	The data you want to write

#### Returns

1 to confirm the operation and 0 for an error



7.57.2.8 `char e_i2cp_write_string ( char device_add, unsigned char write_buffer[], char start_address, char string_length )`

## 7.58 matlab/matlab files/CloseEpuck.m File Reference

### Functions

- `id EpuckPort ()`
- `fclose (EpuckPort)`

### Variables

- `clear EpuckPort`

### 7.58.1 Function Documentation

7.58.1.1 `id EpuckPort ( ) [virtual]`

7.58.1.2 `fclose ( EpuckPort )`

### 7.58.2 Variable Documentation

7.58.2.1 `catch could not open the delete the global variable clear EpuckPort clear global EpuckPort`

## 7.59 matlab/matlab files/EpuckFlush.m File Reference

### Functions

- `id EpuckPort ()`
- `flushinput (EpuckPort)`
- `flushoutput (EpuckPort)`

### Variables

- `function []`
- Matlab probably send old `data` if you don t do this `flush`

## 7.59.1 Function Documentation

7.59.1.1 `id EpuckPort ( ) [virtual]`

7.59.1.2 `flushinput ( EpuckPort )`

7.59.1.3 `flushoutput ( EpuckPort )`

## 7.59.2 Variable Documentation

7.59.2.1 Matlab probably send old data if you don t do this flush

7.59.2.2 `function[]`

### Initial value:

```
= EpuckFlush()
%EPUCKFLUSH Flush the input and output buffer of Matlab
% This function is useful each time you do a send / receive cycle with
% the e-puck. In fact
```

## 7.60 matlab/matlab files/EpuckGetData.m File Reference

### Functions

- `while` (`i~=5`) `c`
- `switch i` case `if` (`c~= 'E'`) `continue`
- end case `if` (`c== 'I'`) `i`
- `elseif` (`c== 'C'`) `i`
- end end end `if` (`receivedFormat== 'I'`) `size`
- `elseif` (`receivedFormat== 'C'`) `size`

### Variables

- `function` [`data`]
- `i = 0`
- `continue`
- `receivedFormat = c`
- `data = fread(EpuckPort,size/2,'int16')`
- end `return`

## 7.60.1 Function Documentation

7.60.1.1 `elseif ( c == 'C' )`

7.60.1.2 `elseif ( receivedFormat == 'C' )`

7.60.1.3 `end case if ( c ~ = 'E' )`

7.60.1.4 `end case if ( c == 'I' )`

7.60.1.5 `end end end if ( receivedFormat == 'I' )`

7.60.1.6 `while ( i ~ = 5 )`

## 7.60.2 Variable Documentation

7.60.2.1 `continue`

7.60.2.2 `data = fread(EpuckPort,size/2,'int16')`

7.60.2.3 `function[data]`

### Initial value:

```
= EpuckGetData()  
global EpuckPort
```

7.60.2.4 `else i = 0`

7.60.2.5 `receivedFormat = c`

7.60.2.6 `end return`

## 7.61 matlab/matlab files/EpuckSendData.m File Reference

### Functions

- `if (strcmp(dataType,'char')) int8(data)`
- `fwrite (EpuckPort, size,'uint16')`
- `fwrite (EpuckPort, data,'int8')`
- `else int16 (data)`
- `fwrite (EpuckPort, 2 *size,'uint16')`
- `fwrite (EpuckPort, data,'int16')`

## Variables

- [function](#) [[data](#)]
- [size](#) = [length](#)([data](#))
- [end](#) [return](#)

### 7.61.1 Function Documentation

7.61.1.1 [fwrite](#) ( [EpuckPort](#) , [size](#) , 'uint16' )

7.61.1.2 [fwrite](#) ( [EpuckPort](#) , [data](#) , 'int8' )

7.61.1.3 [fwrite](#) ( [EpuckPort](#) ,  $2 * \text{size}$  , 'uint16' )

7.61.1.4 [fwrite](#) ( [EpuckPort](#) , [data](#) , 'int16' )

7.61.1.5 [if](#) ( [strcmp](#)([dataType](#), 'char' ) )

7.61.1.6 [else](#) [int16](#) ( [data](#) )

### 7.61.2 Variable Documentation

7.61.2.1 [function](#)[[data](#)]

#### Initial value:

```
= EpuckSendData(data, dataType)
global EpuckPort
```

7.61.2.2 [end](#) [return](#)

7.61.2.3 [size](#) = [length](#)([data](#))

## 7.62 matlab/matlab files/OpenEpuck.m File Reference

### Functions

- [port](#) [EpuckPort](#) ()
- [try](#) [fopen](#) ([EpuckPort](#))

### Variables

- [EpuckPort](#) = [serial](#)([port](#), 'BaudRate', 115200, 'inputBufferSize', 4096, 'OutputBufferSize', 4096, 'ByteOrder', 'littleendian')
- [catch](#) could not open the [port](#)
- [disp](#) [Error](#)
- [disp](#) Could not open serial [port](#) [return](#)

### 7.62.1 Function Documentation

7.62.1.1 `port EpuckPort( ) [virtual]`

7.62.1.2 `try fopen ( EpuckPort )`

### 7.62.2 Variable Documentation

7.62.2.1 `catch could not open the delete the global variable clear EpuckPort clear global EpuckPort = serial(port,'BaudRate', 115200,'inputBufferSize',4096,'OutputBufferSize',4096,'ByteOrder','littleendian')`

7.62.2.2 `disp Error`

7.62.2.3 `catch could not open the port`

7.62.2.4 `disp Could not open serial port return`

## 7.63 matlab/matlab.c File Reference

To communicate with matlab.

```
#include "matlab.h"
#include "../motor_led/e_epuck_ports.h"
#include "../uart/e_uart_char.h"
```

### Functions

- void `e_send_int_to_matlab` (int \*data, int array\_size)  
*The function to send int values to matlab.*
- void `e_send_char_to_matlab` (char \*data, int array\_size)  
*The function to send char values to matlab.*
- int `e_receive_int_from_matlab` (int \*data, int array\_size)  
*The function to receive int values from matlab.*
- int `e_receive_char_from_matlab` (char \*data, int array\_size)  
*The function to receive char values from matlab.*

### 7.63.1 Detailed Description

To communicate with matlab.

This module manage the communication with matlab through bluetooth.

#### Author

Code: Michael Bonani, Jonathan Besuchet, Doc: Jonathan Besuchet

### 7.63.2 Function Documentation

7.63.2.1 `int e_receive_char_from_matlab ( char * data, int array_size )`

The function to receive char values from matlab.

**Parameters**

<i>data</i>	The array of char data you want to fill
<i>array_size</i>	The length of the array

**Returns**

The number of char stored

**7.63.2.2 int e\_receive\_int\_from\_matlab ( int \* *data*, int *array\_size* )**

The function to receive int values from matlab.

**Parameters**

<i>data</i>	The array of int data you want to fill
<i>array_size</i>	The length of the array

**Returns**

The number of int stored

**7.63.2.3 void e\_send\_char\_to\_matlab ( char \* *data*, int *array\_size* )**

The function to send char values to matlab.

**Parameters**

<i>data</i>	The array of char data you want to send
<i>array_size</i>	The length of the array

**7.63.2.4 void e\_send\_int\_to\_matlab ( int \* *data*, int *array\_size* )**

The function to send int values to matlab.

**Parameters**

<i>data</i>	The array of int data you want to send
<i>array_size</i>	The length of the array

**7.64 matlab/matlab.h File Reference**

To communicate with matlab.

## Functions

- void [e\\_send\\_int\\_to\\_matlab](#) (int \**data*, int *array\_size*)  
*The function to send int values to matlab.*
- void [e\\_send\\_char\\_to\\_matlab](#) (char \**data*, int *array\_size*)  
*The function to send char values to matlab.*
- int [e\\_receive\\_int\\_from\\_matlab](#) (int \**data*, int *array\_size*)  
*The function to receive int values from matlab.*
- int [e\\_receive\\_char\\_from\\_matlab](#) (char \**data*, int *array\_size*)  
*The function to receive char values from matlab.*

### 7.64.1 Detailed Description

To communicate with matlab.

This module manage the communication with matlab through bluetooth.

#### Author

Code: Michael Bonani, Jonathan Besuchet, Doc: Jonathan Besuchet

### 7.64.2 Function Documentation

#### 7.64.2.1 int e\_receive\_char\_from\_matlab ( char \* *data*, int *array\_size* )

The function to receive char values from matlab.

##### Parameters

<i>data</i>	The array of char data you want to fill
<i>array_size</i>	The length of the array

##### Returns

The number of char stored

#### 7.64.2.2 int e\_receive\_int\_from\_matlab ( int \* *data*, int *array\_size* )

The function to receive int values from matlab.

##### Parameters

<i>data</i>	The array of int data you want to fill
<i>array_size</i>	The length of the array

**Returns**

The number of int stored

**7.64.2.3 void e\_send\_char\_to\_matlab ( char \* data, int array\_size )**

The function to send char values to matlab.

**Parameters**

<i>data</i>	The array of char data you want to send
<i>array_size</i>	The length of the array

**7.64.2.4 void e\_send\_int\_to\_matlab ( int \* data, int array\_size )**

The function to send int values to matlab.

**Parameters**

<i>data</i>	The array of int data you want to send
<i>array_size</i>	The length of the array

**7.65 motor\_led/advance\_one\_timer/e\_agenda.c File Reference**

Manage the agendas (timer2)

```
#include "e_agenda.h"
#include "../e_epuck_ports.h"
#include <stdlib.h>
```

**Macros**

- `#define EXIT_OK 1`

**Functions**

- void [e\\_start\\_agendas\\_processing](#) (void)  
*Start the agendas processing.*
- void [e\\_end\\_agendas\\_processing](#) (void)  
*Stop all the agendas.*
- int [e\\_activate\\_agenda](#) (void(\*func)(void), int cycle)  
*Activate an agenda.*
- int [e\\_destroy\\_agenda](#) (void(\*func)(void))  
*Destroy an agenda.*



- int `e_set_agenda_cycle` (void(\*func)(void), int cycle)  
*Change the cycle value of an agenda.*
- int `e_reset_agenda` (void(\*func)(void))  
*Reset an agenda's counter.*
- int `e_pause_agenda` (void(\*func)(void))  
*Pause an agenda.*
- int `e_restart_agenda` (void(\*func)(void))  
*Restart an agenda previously paused.*
- void `__attribute__` ((interrupt, auto\_psv))  
*Interrupt from timer2.*

## Variables

- static `Agenda * agenda_list` = 0

### 7.65.1 Detailed Description

Manage the agendas (timer2)

This module manage the agendas with the timer2.

An agenda is a structure made to work as chained list. It contains: the function you want to launch, the time setup between two launching events, a counter to measure the current time, a pointer to the next element of the list.

Each times the timer2 has an interrupt, all the agenda chained list is scanned to look if an agenda has to be treated according to the cycle value and current counter value.

If one (or more) agenda has to be treated, his callback function is launch.

#### Author

Code: Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

### 7.65.2 Macro Definition Documentation

7.65.2.1 `#define EXIT_OK` 1

### 7.65.3 Function Documentation

7.65.3.1 `void __attribute__ ( (interrupt, auto_psv) )`

Interrupt from timer2.

Parse the chained list of agenda.

Increment counter only.

Check if agenda has to be treated according to the cycle value and current counter value.

Do it for number of cycle positive or null.

Check if a service has to be activated.

7.65.3.2 `int e_activate_agenda ( void(*) (void) func, int cycle )`

Activate an agenda.

Activate an agenda and allocate memory for him if there isn't already an agenda with the same callback function (the agenda is active but isn't processed if he has a null cycle value).

## Parameters

<i>func</i>	function called if the cycle value is reached by the counter
<i>cycle</i>	cycle value in millisec/10

## Returns

[EXIT\\_OK](#) if the agenda has been created, exit the programme otherwise

### 7.65.3.3 int e\_destroy\_agenda ( void(\*) (void) *func* )

Destroy an agenda.

Destroy the agenda with a given callback function.

## Parameters

<i>func</i>	function to test
-------------	------------------

## Returns

[EXIT\\_OK](#) if the agenda has been destroyed, [AG\\_NOT\\_FOUND](#) otherwise

### 7.65.3.4 void e\_end\_agendas\_processing ( void )

Stop all the agendas.

Stop all the agendas by disabling Timer2

## Warning

the memory allocated for the agenda isn't freed, use [e\\_destroy\\_agenda](#)(void (\*func)(void)) for that.

## See also

[e\\_destroy\\_agenda](#)

### 7.65.3.5 int e\_pause\_agenda ( void(\*) (void) *func* )

Pause an agenda.

Pause an agenda but do not reset its information.

## Parameters

<i>func</i>	function to pause
-------------	-------------------

**Returns**

[EXIT\\_OK](#) the agenda has been paused, [AG\\_NOT\\_FOUND](#) otherwise

**7.65.3.6 int e\_reset\_agenda ( void(\*) (void) func )**

Reset an agenda's counter.

Reset an agenda's counter with a given callback function.

**Parameters**

<i>func</i>	function to reset
-------------	-------------------

**Returns**

[EXIT\\_OK](#) if the cycle of the agenda has been reseted, [AG\\_NOT\\_FOUND](#) otherwise

**Warning**

This function RESET the agenda, if you just want a pause tell [e\\_pause\\_agenda](#)(void (\*func)(void))

**See also**

[e\\_pause\\_agenda](#)

**7.65.3.7 int e\_restart\_agenda ( void(\*) (void) func )**

Restart an agenda previously paused.

Restart an agenda previously paused.

**Parameters**

<i>func</i>	function to restart
-------------	---------------------

**Returns**

[EXIT\\_OK](#) if he agenda has been restarted, [AG\\_NOT\\_FOUND](#) otherwise

**See also**

[e\\_pause\\_agenda](#)

**7.65.3.8 int e\_set\_agenda\_cycle ( void(\*) (void) func, int cycle )**

Change the cycle value of an agenda.

Change the cycle value of an agenda with a given callback function.

**Parameters**

<i>func</i>	function to test
<i>cycle</i>	new cycle value in millisec/10

**Returns**

[EXIT\\_OK](#) if the cycle of the agenda has been modified, [AG\\_NOT\\_FOUND](#) otherwise

7.65.3.9 void e\_start\_agendas\_processing ( void )

Start the agendas processing.

Start the agendas processing by starting the Timer2.

**7.65.4 Variable Documentation**

7.65.4.1 **Agenda\*** agenda\_list = 0 [static]

pointer on the end of agenda chained list

**7.66 motor\_led/advance\_one\_timer/e\_agenda.h File Reference**

Manage the agendas (timer2)

**Data Structures**

- struct [AgendaType](#)  
*struct Agenda as chained list*

**Macros**

- #define [AG\\_ALREADY\\_CREATED](#) 1
- #define [AG\\_NOT\\_FOUND](#) 2

**Typedefs**

- typedef struct [AgendaType](#) Agenda

## Functions

- void `e_start_agendas_processing` (void)  
*Start the agendas processing.*
- void `e_end_agendas_processing` (void)  
*Stop all the agendas.*
- int `e_activate_agenda` (void(\*func)(void), int cycle)  
*Activate an agenda.*
- int `e_destroy_agenda` (void(\*func)(void))  
*Destroy an agenda.*
- int `e_set_agenda_cycle` (void(\*func)(void), int cycle)  
*Change the cycle value of an agenda.*
- int `e_reset_agenda` (void(\*func)(void))  
*Reset an agenda's counter.*
- int `e_pause_agenda` (void(\*func)(void))  
*Pause an agenda.*
- int `e_restart_agenda` (void(\*func)(void))  
*Restart an agenda previously paused.*

### 7.66.1 Detailed Description

Manage the agendas (timer2)

This module manage the agendas with the timer2.

An agenda is a structure made to work as chained list. It contains: the function you want to launch, the time setup between two launching events, a counter to measure the current time, a pointer to the next element of the list.

Each times the timer2 has an interrupt, all the agenda chained list is scanned to look if an agenda has to be treated according to the cycle value and current counter value.

If one (or more) agenda has to be treated, his callback function is launch.

#### Author

Code: Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

### 7.66.2 Macro Definition Documentation

7.66.2.1 `#define AG_ALREADY_CREATED 1`

7.66.2.2 `#define AG_NOT_FOUND 2`

### 7.66.3 Typedef Documentation

7.66.3.1 `typedef struct AgendaType Agenda`

### 7.66.4 Function Documentation

7.66.4.1 `int e_activate_agenda ( void(*) (void) func, int cycle )`

Activate an agenda.

Activate an agenda and allocate memory for him if there isn't already an agenda with the same callback function (the agenda is active but isn't processed if he has a null cycle value).

## Parameters

<i>func</i>	function called if the cycle value is reached by the counter
<i>cycle</i>	cycle value in millisec/10

## Returns

[EXIT\\_OK](#) if the agenda has been created, exit the programme otherwise

7.66.4.2 `int e_destroy_agenda ( void(*) (void) func )`

Destroy an agenda.

Destroy the agenda with a given callback function.

## Parameters

<i>func</i>	function to test
-------------	------------------

## Returns

[EXIT\\_OK](#) if the agenda has been destroyed, [AG\\_NOT\\_FOUND](#) otherwise

Destroy the agenda with a given callback function

## Parameters

<i>func</i>	function to test
-------------	------------------

## Returns

int return the success of the destruction (EXIT\_OK for successfull, AG\_NOT\_FOUND for unsuccessful).

7.66.4.3 `void e_end_agendas_processing ( void )`

Stop all the agendas.

Stop all the agendas by disabling Timer2

## Warning

the memory allocated for the agenda isn't freed, use [e\\_destroy\\_agenda](#)(void (\*func)(void)) for that.

## See also

[e\\_destroy\\_agenda](#)

7.66.4.4 `int e_pause_agenda ( void(*) (void) func )`

Pause an agenda.

Pause an agenda but do not reset its information.

**Parameters**

<i>func</i>	function to pause
-------------	-------------------

**Returns**

[EXIT\\_OK](#) the agenda has been paused, [AG\\_NOT\\_FOUND](#) otherwise

Pause an agenda but do not reset its information.

**Parameters**

<i>func</i>	function to pause
-------------	-------------------

**7.66.4.5 int e\_reset\_agenda ( void(\*) (void) *func* )**

Reset an agenda's counter.

Reset an agenda's counter with a given callback function.

**Parameters**

<i>func</i>	function to reset
-------------	-------------------

**Returns**

[EXIT\\_OK](#) if the cycle of the agenda has been reseted, [AG\\_NOT\\_FOUND](#) otherwise

**Warning**

This function RESET the agenda, if you just want a pause tell [e\\_pause\\_agenda](#)(void (\*func)(void))

**See also**

[e\\_pause\\_agenda](#)

Reset an agenda's counter.

Reset an agenda's counter with a given callback function

**Parameters**

<i>func</i>	function to reset
-------------	-------------------



#### 7.66.4.6 int e\_restart\_agenda ( void(\*) (void) func )

Restart an agenda previously paused.

Restart an agenda previously paused.

##### Parameters

<i>func</i>	function to restart
-------------	---------------------

##### Returns

[EXIT\\_OK](#) if the agenda has been restarted, [AG\\_NOT\\_FOUND](#) otherwise

##### See also

[e\\_pause\\_agenda](#)

Restart an agenda previously paused

##### Parameters

<i>func</i>	function to restart
-------------	---------------------

#### 7.66.4.7 int e\_set\_agenda\_cycle ( void(\*) (void) func, int cycle )

Change the cycle value of an agenda.

Change the cycle value of an agenda with a given callback function.

##### Parameters

<i>func</i>	function to test
<i>cycle</i>	new cycle value in millisec/10

##### Returns

[EXIT\\_OK](#) if the cycle of the agenda has been modified, [AG\\_NOT\\_FOUND](#) otherwise

#### 7.66.4.8 void e\_start\_agendas\_processing ( void )

Start the agendas processing.

Start the agendas processing by starting the Timer2.

Start the agendas processing.

Start the agendas processing by initialising the accounting structures.

Don't activate any timer which is done by e\_start\_timer\_processing.

See also

[e\\_start\\_timer\\_processing](#)

## 7.67 motor\_led/advance\_one\_timer/e\_led.c File Reference

Manage the LEDs with blinking possibility (timer2).

```
#include "../e_epuck_ports.h"  
#include "e_agenda.h"
```

### Macros

- `#define LED_EFFECTS`  
*For the preprocessor. Comment if you want not to use the LED effects.*

### Functions

- void [e\\_set\\_led](#) (unsigned int led\_number, unsigned int value)  
*turn on/off the specified LED*
- void [e\\_led\\_clear](#) (void)  
*turn off the 8 LEDs*
- void [e\\_set\\_body\\_led](#) (unsigned int value)  
*turn on/off the body LED*
- void [e\\_set\\_front\\_led](#) (unsigned int value)  
*turn on/off the front LED*
- void [e\\_blink\\_led](#) (void)  
*Change the state of all LED.*
- void [e\\_blink\\_led0](#) (void)  
*Change the state of LED0.*
- void [e\\_blink\\_led1](#) (void)  
*Change the state of LED1.*
- void [e\\_blink\\_led2](#) (void)  
*Change the state of LED2.*
- void [e\\_blink\\_led3](#) (void)  
*Change the state of LED3.*
- void [e\\_blink\\_led4](#) (void)  
*Change the state of LED4.*
- void [e\\_blink\\_led5](#) (void)  
*Change the state of LED5.*
- void [e\\_blink\\_led6](#) (void)  
*Change the state of LED6.*
- void [e\\_blink\\_led7](#) (void)  
*Change the state of LED7.*
- void [e\\_start\\_led\\_blinking](#) (int cycle)  
*Start blinking all LED.*
- void [e\\_stop\\_led\\_blinking](#) (void)  
*Stop blinking all LED.*

- void [e\\_set\\_blinking\\_cycle](#) (int cycle)  
*Change the blinking speed.*
- void [snake\\_led](#) (void)  
*One led is on and turn clockwise.*
- void [flow\\_led](#) (void)  
*The leds go on from the front to the back and go off from the front to the back, etc.*
- void [k2000\\_led](#) (void)  
*The K2000 effect.*
- void [right\\_led](#) (void)  
*The right LED are indicating the right side.*
- void [left\\_led](#) (void)  
*The left LED are indicating the left side.*

### 7.67.1 Detailed Description

Manage the LEDs with blinking possibility (timer2).

Here we use the agenda solution to make the LED blinking.

A little exemple for LEDs blinking with agenda (all LEDs blink with 100ms delay)

#### Warning

this program uses the [e\\_blink\\_led\(void\)](#) function.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_led.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    e_init_port();
    e_activate_agenda(e_blink_led, 1000); //blink with 100ms
    e_start_agendas_processing();
    while(1) {}
}
```

#### See also

[e\\_agenda.h](#)

#### Author

Code: Francesco Mondada, Lucas Meier, Jonathan Besuchet  
Doc: Jonathan Besuchet

### 7.67.2 Macro Definition Documentation

#### 7.67.2.1 #define LED\_EFFECTS

For the preprocessor. Comment if you want not to use the LED effects.

### 7.67.3 Function Documentation

#### 7.67.3.1 void e\_blink\_led ( void )

Change the state of all LED.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.67.3.2 void e\_blink\_led0 ( void )

Change the state of LED0.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.67.3.3 void e\_blink\_led1 ( void )

Change the state of LED1.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.67.3.4 void e\_blink\_led2 ( void )

Change the state of LED2.

Callback function for an agenda.

See also

[AgendaType](#)

7.67.3.5 void e\_blink\_led3 ( void )

Change the state of LED3.

Callback function for an agenda.

See also

[AgendaType](#)

7.67.3.6 void e\_blink\_led4 ( void )

Change the state of LED4.

Callback function for an agenda.

See also

[AgendaType](#)

7.67.3.7 void e\_blink\_led5 ( void )

Change the state of LED5.

Callback function for an agenda.

See also

[AgendaType](#)

7.67.3.8 void e\_blink\_led6 ( void )

Change the state of LED6.

Callback function for an agenda.

See also

[AgendaType](#)

7.67.3.9 void e\_blink\_led7 ( void )

Change the state of LED7.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.67.3.10 void e\_led\_clear ( void )

turn off the 8 LEDs

The e-puck has 8 green LEDs. This function turn all off.

#### Warning

this function doesn't turn off "body LED" and "front LED".

### 7.67.3.11 void e\_set\_blinking\_cycle ( int cycle )

Change the blinking speed.

This function use [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)

#### Parameters

<i>cycle</i>	the number of cycle we wait before launching <a href="#">e_blink_led</a> (void)"
--------------	--

#### See also

[e\\_blink\\_led](#), [e\\_set\\_agenda\\_cycle](#)

### 7.67.3.12 void e\_set\_body\_led ( unsigned int value )

turn on/off the body LED

The e-puck has a green LED that illuminate his body. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

### 7.67.3.13 void e\_set\_front\_led ( unsigned int value )

turn on/off the front LED

The e-puck has a red LED in the front. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

7.67.3.14 void e\_set\_led ( unsigned int *led\_number*, unsigned int *value* )

turn on/off the specified LED

The e-puck has 8 green LEDs. With this function, you can change the state of these LEDs.

Parameters

<i>led_number</i>	between 0 and 7
<i>value</i>	0 (off), 1 (on) otherwise change the state

Warning

if *led\_number* is other than 0-7, all leds are set to the indicated value.

7.67.3.15 void e\_start\_led\_blinking ( int *cycle* )

Start blinking all LED.

Parameters

<i>cycle</i>	the number of cycle we wait before launching <a href="#">e_blink_led(void)</a>
--------------	--

See also

[e\\_blink\\_led](#), [e\\_activate\\_agenda](#)

7.67.3.16 void e\_stop\_led\_blinking ( void )

Stop blinking all LED.

This function use [e\\_destroy\\_agenda\(void \(\\*func\)\(void\)\)](#)

See also

[e\\_destroy\\_agenda](#)

7.67.3.17 void flow\_led ( void )

The leds go on from the front to the back and go off from the front to the back, etc.

7.67.3.18 void k2000\_led ( void )

The K2000 effect.

### 7.67.3.19 void left\_led ( void )

The left LED are indicating the left side.

### 7.67.3.20 void right\_led ( void )

The right LED are indicating the right side.

### 7.67.3.21 void snake\_led ( void )

One led is on and turn clockwise.

## 7.68 motor\_led/advance\_one\_timer/fast\_agenda/e\_led.c File Reference

Manage the LEDs with blinking possibility (timer1, 2, 3).

```
#include "../e_epuck_ports.h"
#include "e_agenda_fast.h"
```

### Functions

- void [e\\_set\\_led](#) (unsigned int led\_number, unsigned int value)  
*turn on/off the specified LED*
- void [e\\_led\\_clear](#) (void)  
*turn off the 8 LEDs*
- void [e\\_set\\_body\\_led](#) (unsigned int value)  
*turn on/off the body LED*
- void [e\\_set\\_front\\_led](#) (unsigned int value)  
*turn on/off the front LED*
- void [e\\_blink\\_led](#) (void)  
*Change the state of all LED.*
- void [e\\_blink\\_led0](#) (void)  
*Change the state of LED0.*
- void [e\\_blink\\_led1](#) (void)  
*Change the state of LED1.*
- void [e\\_blink\\_led2](#) (void)  
*Change the state of LED2.*
- void [e\\_blink\\_led3](#) (void)  
*Change the state of LED3.*
- void [e\\_blink\\_led4](#) (void)  
*Change the state of LED4.*
- void [e\\_blink\\_led5](#) (void)  
*Change the state of LED5.*
- void [e\\_blink\\_led6](#) (void)  
*Change the state of LED6.*
- void [e\\_blink\\_led7](#) (void)  
*Change the state of LED7.*
- void [e\\_start\\_led\\_blinking](#) (int cycle)  
*Start blinking all LED.*
- void [e\\_stop\\_led\\_blinking](#) (void)  
*Stop blinking all LED.*
- void [e\\_set\\_blinking\\_cycle](#) (int cycle)  
*Change the blinking speed.*



### 7.68.1 Detailed Description

Manage the LEDs with blinking possibility (timer1, 2, 3).

Here we use the fast agenda solution to make the LED blinking.

See also

[e\\_agenda.h](#)

Author

Code: Francesco Mondada

Doc: Jonathan Besuchet

### 7.68.2 Function Documentation

#### 7.68.2.1 void e\_blink\_led ( void )

Change the state of all LED.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.2 void e\_blink\_led0 ( void )

Change the state of LED0.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.3 void e\_blink\_led1 ( void )

Change the state of LED1.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.4 void e\_blink\_led2 ( void )

Change the state of LED2.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.5 void e\_blink\_led3 ( void )

Change the state of LED3.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.6 void e\_blink\_led4 ( void )

Change the state of LED4.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.7 void e\_blink\_led5 ( void )

Change the state of LED5.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.68.2.8 void e\_blink\_led6 ( void )

Change the state of LED6.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.68.2.9 void e\_blink\_led7 ( void )

Change the state of LED7.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.68.2.10 void e\_led\_clear ( void )

turn off the 8 LEDs

The e-puck has 8 green LEDs. This function turn all off.

**Warning**

this function doesn't turn off "body LED" and "front LED".

### 7.68.2.11 void e\_set\_blinking\_cycle ( int cycle )

Change the blinking speed.

This function use e\_set\_agenda\_cycle(void (\*func)(void), int cycle)

**Parameters**

<i>cycle</i>	the number of cycle we wait before launching "e_blink_led()"
--------------	--

See also

[e\\_blink\\_led](#), [e\\_set\\_agenda\\_cycle](#)

### 7.68.2.12 void e\_set\_body\_led ( unsigned int value )

turn on/off the body LED

The e-puck has a green LED that illuminate his body. With this function, you can change the state of these LED.

**Parameters**

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

### 7.68.2.13 void e\_set\_front\_led ( unsigned int *value* )

turn on/off the front LED

The e-puck has a red LED in the front. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

### 7.68.2.14 void e\_set\_led ( unsigned int *led\_number*, unsigned int *value* )

turn on/off the specified LED

The e-puck has 8 green LEDs. With this function, you can change the state of these LEDs.

#### Parameters

<i>led_number</i>	between 0 and 7
<i>value</i>	0 (off), 1 (on) otherwise change the state

#### Warning

if *led\_number* is other than 0-7, all leds are set to the indicated value.

### 7.68.2.15 void e\_start\_led\_blinking ( int *cycle* )

Start blinking all LED.

#### Parameters

<i>cycle</i>	the number of cycle we wait before launching "e_blink_led()"
--------------	--

#### See also

[e\\_blink\\_led](#), [e\\_activate\\_agenda](#)

### 7.68.2.16 void e\_stop\_led\_blinking ( void )

Stop blinking all LED.

This function use `e_destroy_agenda(void (*func)(void))`

#### See also

[e\\_destroy\\_agenda](#)

## 7.69 motor\_led/e\_led.c File Reference

Manage the LEDs.

A little exemple for the LEDs (all the LEDs are blinking)

```
#include "e_epuck_ports.h"
#include "e_led.h"
```

### Functions

- void `e_set_led` (unsigned int led\_number, unsigned int value)  
*turn on/off the specified LED*
- void `e_set_body_led` (unsigned int value)  
*turn on/off the body LED*
- void `e_set_front_led` (unsigned int value)  
*turn on/off the front LED*
- void `e_led_clear` (void)  
*turn off the 8 LEDs*

### 7.69.1 Detailed Description

Manage the LEDs.

A little exemple for the LEDs (all the LEDs are blinking)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/e_led.h>

int main(void)
{
    e_init_port ();
    while(1)
    {
        long i;
        for(i=0; i<5000000; i++)
            asm("NOP");
        e_set_led(8, 2); //switch the state of all leds
    }
}
```

### Author

Code: Michael Bonani, Francesco Mondada, Davis Dadie  
Doc: Jonathan Besuchet

### 7.69.2 Function Documentation

#### 7.69.2.1 void e\_led\_clear ( void )

turn off the 8 LEDs

The e-puck has 8 green LEDs. This function turn all off.

#### Warning

this function doesn't turn off "body LED" and "front LED".

7.69.2.2 void e\_set\_body\_led ( unsigned int *value* )

turn on/off the body LED

The e-puck has a green LED that illuminate his body. With this function, you can change the state of this LED.

## Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

7.69.2.3 void e\_set\_front\_led ( unsigned int *value* )

turn on/off the front LED

The e-puck has a red LED in the front. With this function, you can change the state of this LED.

## Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

7.69.2.4 void e\_set\_led ( unsigned int *led\_number*, unsigned int *value* )

turn on/off the specified LED

The e-puck has 8 green LEDs. With this function, you can change the state of these LEDs.

## Parameters

<i>led_number</i>	between 0 and 7
<i>value</i>	0 (off), 1 (on) otherwise change the state

## Warning

if *led\_number* is other than 0-7, all leds are set to the indicated value.

## 7.70 motor\_led/advance\_one\_timer/e\_led.h File Reference

Manage the LEDs with blinking possibility (timer2).

## Functions

- void [e\\_set\\_led](#) (unsigned int *led\_number*, unsigned int *value*)  
*turn on/off the specified LED*
- void [e\\_led\\_clear](#) (void)  
*turn off the 8 LEDs*
- void [e\\_blink\\_led](#) (void)  
*Change the state of all LED.*
- void [e\\_blink\\_led0](#) (void)  
*Change the state of LED0.*
- void [e\\_blink\\_led1](#) (void)

- Change the state of LED1.*

  - void `e_blink_led2` (void)
- Change the state of LED2.*

  - void `e_blink_led3` (void)
- Change the state of LED3.*

  - void `e_blink_led4` (void)
- Change the state of LED4.*

  - void `e_blink_led5` (void)
- Change the state of LED5.*

  - void `e_blink_led6` (void)
- Change the state of LED6.*

  - void `e_blink_led7` (void)
- Change the state of LED7.*

  - void `e_set_body_led` (unsigned int value)

*turn on/off the body LED*
- turn on/off the front LED*

  - void `e_set_front_led` (unsigned int value)
- Start blinking all LED.*

  - void `e_start_led_blinking` (int cycle)
- Stop blinking all LED.*

  - void `e_stop_led_blinking` (void)
- The leds go on from the front to the back and go off from the front to the back, etc.*

  - void `flow_led` (void)
- One led is on and turn clockwise.*

  - void `snake_led` (void)
- The K2000 effect.*

  - void `k2000_led` (void)
- The right LED are indicating the right side.*

  - void `right_led` (void)
- The left LED are indicating the left side.*

  - void `left_led` (void)

### 7.70.1 Detailed Description

Manage the LEDs with blinking possibility (timer2).

Here we use the agenda solution to make the LED blinking.

A little exemple for LEDs blinking with agenda (all LEDs blink with 100ms delay)

#### Warning

this program uses the `e_blink_led(void)` function.

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_led.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    e_init_port();
    e_activate_agenda(e_blink_led, 1000); //blink with 100ms
    e_start_agendas_processing();
    while(1) {}
}
```



See also

[e\\_agenda.h](#)

Author

Code: Francesco Mondada, Lucas Meier, Jonathan Besuchet

Doc: Jonathan Besuchet

## 7.70.2 Function Documentation

### 7.70.2.1 void e\_blink\_led ( void )

Change the state of all LED.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.70.2.2 void e\_blink\_led0 ( void )

Change the state of LED0.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.70.2.3 void e\_blink\_led1 ( void )

Change the state of LED1.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.70.2.4 void e\_blink\_led2 ( void )

Change the state of LED2.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.70.2.5 void e\_blink\_led3 ( void )

Change the state of LED3.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.70.2.6 void e\_blink\_led4 ( void )

Change the state of LED4.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.70.2.7 void e\_blink\_led5 ( void )

Change the state of LED5.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.70.2.8 void e\_blink\_led6 ( void )

Change the state of LED6.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.70.2.9 void e\_blink\_led7 ( void )

Change the state of LED7.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.70.2.10 void e\_led\_clear ( void )

turn off the 8 LEDs

The e-puck has 8 green LEDs. This function turn all off.

#### Warning

this function doesn't turn off "body LED" and "front LED".

#### 7.70.2.11 void e\_set\_body\_led ( unsigned int *value* )

turn on/off the body LED

The e-puck has a green LED that illuminate his body. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

The e-puck has a green LED that illuminate his body. With this function, you can change the state of this LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

#### 7.70.2.12 void e\_set\_front\_led ( unsigned int *value* )

turn on/off the front LED

The e-puck has a red LED in the front. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

The e-puck has a red LED in the front. With this function, you can change the state of this LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

#### 7.70.2.13 void e\_set\_led ( unsigned int *led\_number*, unsigned int *value* )

turn on/off the specified LED

The e-puck has 8 green LEDs. With this function, you can change the state of these LEDs.

## Parameters

<i>led_number</i>	between 0 and 7
<i>value</i>	0 (off), 1 (on) otherwise change the state

## Warning

if led\_number is other than 0-7, all leds are set to the indicated value.

## 7.70.2.14 void e\_start\_led\_blinking ( int cycle )

Start blinking all LED.

## Parameters

<i>cycle</i>	the number of cycle we wait before launching <a href="#">e_blink_led(void)</a>
--------------	--

## See also

[e\\_blink\\_led](#), [e\\_activate\\_agenda](#)

## Parameters

<i>cycle</i>	the number of cycle we wait before launching " <a href="#">e_blink_led()</a> "
--------------	--

## See also

[e\\_blink\\_led](#), [e\\_activate\\_agenda](#)

## 7.70.2.15 void e\_stop\_led\_blinking ( void )

Stop blinking all LED.

This function use [e\\_destroy\\_agenda\(void \(\\*func\)\(void\)\)](#)

## See also

[e\\_destroy\\_agenda](#)

This function use [e\\_destroy\\_agenda\(void \(\\*func\)\(void\)\)](#)

## See also

[e\\_destroy\\_agenda](#)

#### 7.70.2.16 void flow\_led ( void )

The leds go on from the front to the back and go off from the front to the back, etc.

#### 7.70.2.17 void k2000\_led ( void )

The K2000 effect.

#### 7.70.2.18 void left\_led ( void )

The left LED are indicating the left side.

#### 7.70.2.19 void right\_led ( void )

The right LED are indicating the right side.

#### 7.70.2.20 void snake\_led ( void )

One led is on and turn clockwise.

## 7.71 motor\_led/advance\_one\_timer/fast\_agenda/e\_led.h File Reference

Manage the LEDs with blinking possibility (timer1, 2, 3).

### Functions

- void [e\\_set\\_led](#) (unsigned int led\_number, unsigned int value)  
*turn on/off the specified LED*
- void [e\\_led\\_clear](#) (void)  
*turn off the 8 LEDs*
- void [e\\_blink\\_led](#) (void)  
*Change the state of all LED.*
- void [e\\_blink\\_led0](#) (void)  
*Change the state of LED0.*
- void [e\\_blink\\_led1](#) (void)  
*Change the state of LED1.*
- void [e\\_blink\\_led2](#) (void)  
*Change the state of LED2.*
- void [e\\_blink\\_led3](#) (void)  
*Change the state of LED3.*
- void [e\\_blink\\_led4](#) (void)  
*Change the state of LED4.*
- void [e\\_blink\\_led5](#) (void)  
*Change the state of LED5.*

- void [e\\_blink\\_led6](#) (void)  
*Change the state of LED6.*
- void [e\\_blink\\_led7](#) (void)  
*Change the state of LED7.*
- void [e\\_set\\_body\\_led](#) (unsigned int value)  
*turn on/off the body LED*
- void [e\\_set\\_front\\_led](#) (unsigned int value)  
*turn on/off the front LED*
- void [e\\_start\\_led\\_blinking](#) (int cycle)  
*Start blinking all LED.*
- void [e\\_stop\\_led\\_blinking](#) (void)  
*Stop blinking all LED.*

### 7.71.1 Detailed Description

Manage the LEDs with blinking possibility (timer1, 2, 3).

Here we use the fast agenda solution to make the LED blinking.

See also

[e\\_agenda.h](#)

Author

Code: Francesco Mondada

Doc: Jonathan Besuchet

### 7.71.2 Function Documentation

#### 7.71.2.1 void [e\\_blink\\_led](#) ( void )

Change the state of all LED.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.71.2.2 void [e\\_blink\\_led0](#) ( void )

Change the state of LED0.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.71.2.3 void e\_blink\_led1 ( void )

Change the state of LED1.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.71.2.4 void e\_blink\_led2 ( void )

Change the state of LED2.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.71.2.5 void e\_blink\_led3 ( void )

Change the state of LED3.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.71.2.6 void e\_blink\_led4 ( void )

Change the state of LED4.

Callback function for an agenda.

See also

[AgendaType](#)

### 7.71.2.7 void e\_blink\_led5 ( void )

Change the state of LED5.

Callback function for an agenda.

See also

[AgendaType](#)



#### 7.71.2.8 void e\_blink\_led6 ( void )

Change the state of LED6.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.71.2.9 void e\_blink\_led7 ( void )

Change the state of LED7.

Callback function for an agenda.

See also

[AgendaType](#)

#### 7.71.2.10 void e\_led\_clear ( void )

turn off the 8 LEDs

The e-puck has 8 green LEDs. This function turn all off.

**Warning**

this function doesn't turn off "body LED" and "front LED".

#### 7.71.2.11 void e\_set\_body\_led ( unsigned int *value* )

turn on/off the body LED

The e-puck has a green LED that illuminate his body. With this function, you can change the state of these LED.

**Parameters**

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

The e-puck has a green LED that illuminate his body. With this function, you can change the state of this LED.

**Parameters**

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

### 7.71.2.12 void e\_set\_front\_led ( unsigned int *value* )

turn on/off the front LED

The e-puck has a red LED in the front. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

The e-puck has a red LED in the front. With this function, you can change the state of this LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

### 7.71.2.13 void e\_set\_led ( unsigned int *led\_number*, unsigned int *value* )

turn on/off the specified LED

The e-puck has 8 green LEDs. With this function, you can change the state of these LEDs.

#### Parameters

<i>led_number</i>	between 0 and 7
<i>value</i>	0 (off), 1 (on) otherwise change the state

#### Warning

if *led\_number* is other than 0-7, all leds are set to the indicated value.

### 7.71.2.14 void e\_start\_led\_blinking ( int *cycle* )

Start blinking all LED.

#### Parameters

<i>cycle</i>	the number of cycle we wait before launching <a href="#">e_blink_led(void)</a>
--------------	--

#### See also

[e\\_blink\\_led](#), [e\\_activate\\_agenda](#)

#### Parameters

<i>cycle</i>	the number of cycle we wait before launching "e_blink_led()"
--------------	--

See also

[e\\_blink\\_led](#), [e\\_activate\\_agenda](#)

#### 7.71.2.15 void e\_stop\_led\_blinking ( void )

Stop blinking all LED.

This function use [e\\_destroy\\_agenda](#)(void (\*func)(void))

See also

[e\\_destroy\\_agenda](#)

This function use [e\\_destroy\\_agenda](#)(void (\*func)(void))

See also

[e\\_destroy\\_agenda](#)

## 7.72 motor\_led/e\_led.h File Reference

Manage the LEDs.

A little exemple for the LEDs (all the LEDs are blinking)

### Functions

- void [e\\_set\\_led](#) (unsigned int led\_number, unsigned int value)  
*turn on/off the specified LED*
- void [e\\_led\\_clear](#) (void)  
*turn off the 8 LEDs*
- void [e\\_set\\_body\\_led](#) (unsigned int value)  
*turn on/off the body LED*
- void [e\\_set\\_front\\_led](#) (unsigned int value)  
*turn on/off the front LED*

### 7.72.1 Detailed Description

Manage the LEDs.

A little exemple for the LEDs (all the LEDs are blinking)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/e_led.h>

int main(void)
{
    e_init_port ();
    while(1)
    {
        long i;
        for(i=0; i<500000; i++)
            asm("NOP");
        e_set_led(8, 2); //switch the state of all leds
    }
}
```

### Author

Code: Michael Bonani, Francesco Mondada, Davis Dadie  
Doc: Jonathan Besuchet

## 7.72.2 Function Documentation

### 7.72.2.1 void e\_led\_clear ( void )

turn off the 8 LEDs

The e-puck has 8 green LEDs. This function turn all off.

#### Warning

this function doesn't turn off "body LED" and "front LED".

### 7.72.2.2 void e\_set\_body\_led ( unsigned int *value* )

turn on/off the body LED

The e-puck has a green LED that illuminate his body. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

The e-puck has a green LED that illuminate his body. With this function, you can change the state of this LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

### 7.72.2.3 void e\_set\_front\_led ( unsigned int *value* )

turn on/off the front LED

The e-puck has a red LED in the front. With this function, you can change the state of these LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

The e-puck has a red LED in the front. With this function, you can change the state of this LED.

#### Parameters

<i>value</i>	0 (off), 1 (on) otherwise change the state
--------------	--

## 7.72.2.4 void e\_set\_led ( unsigned int led\_number, unsigned int value )

turn on/off the specified LED

The e-puck has 8 green LEDs. With this function, you can change the state of these LEDs.

## Parameters

<i>led_number</i>	between 0 and 7
<i>value</i>	0 (off), 1 (on) otherwise change the state

## Warning

if led\_number is other than 0-7, all leds are set to the indicated value.

## 7.73 motor\_led/advance\_one\_timer/e\_motors.c File Reference

Manage the motors (with timer2)

```
#include "../e_epuck_ports.h"
#include "e_agenda.h"
#include <stdlib.h>
```

## Macros

- #define POWERSAVE
- #define TRESHV 650
- #define MAXV 601

## Functions

- void run\_left\_motor (void)
- void run\_right\_motor (void)
- void e\_init\_motors (void)
  - Initialize the motors's agendas.*
- void e\_set\_speed\_left (int motor\_speed)
  - Manage the left motor speed.*
- void e\_set\_speed\_right (int motor\_speed)
  - Manage the right motor speed.*
- void e\_set\_speed (int linear\_speed, int angular\_speed)
  - Manage linear/angular speed.*
- int e\_get\_steps\_left ()
  - Give the number of left motor steps.*
- void e\_set\_steps\_left (int set\_steps)
  - Set the number of left motor steps.*
- int e\_get\_steps\_right ()
  - Give the number of right motor steps.*
- void e\_set\_steps\_right (int set\_steps)
  - Set the number of right motor steps.*

## Variables

- static int [left\\_speed](#) = 0
- static int [right\\_speed](#) = 0
- static int [left\\_motor\\_phase](#) =0
- static int [right\\_motor\\_phase](#) =0
- static int [nbr\\_steps\\_left](#) =0
- static int [nbr\\_steps\\_right](#) =0

### 7.73.1 Detailed Description

Manage the motors (with timer2)

This module manage the motors with the agenda solution (timer2).

A little exemple to use the motors with agenda (e-puck turn on himself)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_motors.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    e_init_port();
    e_init_motors();
    e_set_speed(-500, 500);
    e_start_agendas_processing();
    while(1) {}
}
```

#### See also

[e\\_agenda.h](#)

#### Author

Code: Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

### 7.73.2 Macro Definition Documentation

7.73.2.1 `#define MAXV 601`

7.73.2.2 `#define POWERSAVE`

7.73.2.3 `#define TRESHV 650`

### 7.73.3 Function Documentation

7.73.3.1 `int e_get_steps_left ( void )`

Give the number of left motor steps.

#### Returns

The number of phases steps made since the left motor is running.

## 7.73.3.2 int e\_get\_steps\_right ( void )

Give the number of right motor steps.

## Returns

The number of phases steps made since the right motor is running.

## 7.73.3.3 void e\_init\_motors ( void )

Initialize the motors's agendas.

This function initialize the agendas used by the motors. In fact it call [e\\_activate\\_agenda](#)(void (\*func)(void), int cycle) function.

## See also

[e\\_activate\\_agenda](#)

7.73.3.4 void e\_set\_speed ( int *linear\_speed*, int *angular\_speed* )

Manage linear/angular speed.

This function manage the speed of the motors according to the desired linear and angular speed.

## Parameters

<i>linear_speed</i>	the speed in the axis of e-puck
<i>angular_speed</i>	the rotation speed (trigonometric)

7.73.3.5 void e\_set\_speed\_left ( int *motor\_speed* )

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

### 7.73.3.6 void e\_set\_speed\_right ( int motor\_speed )

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### See also

[e\\_set\\_agenda\\_cycle](#)

### 7.73.3.7 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

#### Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

### 7.73.3.8 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

#### Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

### 7.73.3.9 void run\_left\_motor ( void )

Change left motor phase according to the left\_speed sign.

### 7.73.3.10 void run\_right\_motor ( void )

Change right motor phase according to the right\_speed sign



## 7.73.4 Variable Documentation

7.73.4.1 `int left_motor_phase = 0` [static]

7.73.4.2 `int left_speed = 0` [static]

7.73.4.3 `int nbr_steps_left = 0` [static]

7.73.4.4 `int nbr_steps_right = 0` [static]

7.73.4.5 `int right_motor_phase = 0` [static]

7.73.4.6 `int right_speed = 0` [static]

## 7.74 motor\_led/advance\_one\_timer/fast\_agenda/e\_motors.c File Reference

Manage the motors (with timer1, 2, 3)

```
#include "../e_epuck_ports.h"
#include "e_agenda_fast.h"
#include <stdlib.h>
```

### Macros

- #define `POWERSAVE`
- #define `TRESHV` 650
- #define `MAXV` 601

### Functions

- void `run_left_motor` (void)
- void `run_right_motor` (void)
- void `e_init_motors` (void)  
*Initialize the motors's agendas.*
- void `e_set_speed_left` (int motor\_speed)  
*Manage the left motor speed.*
- void `e_set_speed_right` (int motor\_speed)  
*Manage the right motor speed.*
- void `e_set_speed` (int linear\_speed, int angular\_speed)  
*Manage linear/angular speed.*
- int `e_get_steps_left` ()  
*Give the number of left motor steps.*
- void `e_set_steps_left` (int set\_steps)  
*Set the number of left motor steps.*
- int `e_get_steps_right` ()  
*Give the number of right motor steps.*
- void `e_set_steps_right` (int set\_steps)  
*Set the number of right motor steps.*

## Variables

- static int `left_speed` = 0
- static int `right_speed` = 0
- static int `left_motor_phase` =0
- static int `right_motor_phase` =0
- static int `nbr_steps_left` =0
- static int `nbr_steps_right` =0

### 7.74.1 Detailed Description

Manage the motors (with timer1, 2, 3)

This module manage the motors with the fast agenda solution (timer1, 2, 3).

See also

[e\\_agenda.h](#)

Author

Code: Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

### 7.74.2 Macro Definition Documentation

7.74.2.1 `#define MAXV 601`

7.74.2.2 `#define POWERSAVE`

7.74.2.3 `#define TRESHV 650`

### 7.74.3 Function Documentation

7.74.3.1 `int e_get_steps_left ( void )`

Give the number of left motor steps.

Returns

The number of phases steps made since the left motor is running.

7.74.3.2 `int e_get_steps_right ( void )`

Give the number of right motor steps.

Returns

The number of phases steps made since the right motor is running.

## 7.74.3.3 void e\_init\_motors ( void )

Initialize the motors's agendas.

This function initialize the agendas used by the motors. In fact it call "e\_activate\_agenda(void (\*func)(void), int cycle)" function.

See also

[e\\_activate\\_agenda](#)

7.74.3.4 void e\_set\_speed ( int *linear\_speed*, int *angular\_speed* )

Manage linear/angular speed.

This function manage the speed of the motors according to the desired linear and angular speed.

Parameters

<i>linear_speed</i>	the speed in the axis of e-puck
<i>angular_speed</i>	the rotation speed (trigonometric)

7.74.3.5 void e\_set\_speed\_left ( int *motor\_speed* )

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

See also

[e\\_set\\_agenda\\_cycle](#)

7.74.3.6 void e\_set\_speed\_right ( int *motor\_speed* )

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

## 7.74.3.7 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

## 7.74.3.8 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

## 7.74.3.9 void run\_left\_motor ( void )

Change left motor phase according to the left\_speed signe.

## 7.74.3.10 void run\_right\_motor ( void )

Change right motor phase according to the right\_speed signe

## 7.74.4 Variable Documentation

7.74.4.1 int left\_motor\_phase =0 [static]

7.74.4.2 int left\_speed = 0 [static]

7.74.4.3 int nbr\_steps\_left =0 [static]

7.74.4.4 int nbr\_steps\_right =0 [static]

7.74.4.5 `int right_motor_phase = 0` [static]

7.74.4.6 `int right_speed = 0` [static]

## 7.75 motor\_led/e\_motors.c File Reference

Manage the motors (with timer 4 and 5).

```
#include <stdlib.h>
#include "e_epuck_ports.h"
#include "e_init_port.h"
#include "e_motors.h"
```

### Functions

- void `__attribute__` ((interrupt, auto\_psv, shadow))
- void `e_init_motors` (void)  
*Initialize the motors's ports.*
- void `e_set_speed_left` (int motor\_speed)  
*Manage the left speed.*
- void `e_set_speed_right` (int motor\_speed)  
*Manage the right speed.*
- int `e_get_steps_left` (void)  
*Give the number of left motor steps.*
- void `e_set_steps_left` (int set\_steps)  
*Set the number of left motor steps.*
- int `e_get_steps_right` (void)  
*Give the number of right motor steps.*
- void `e_set_steps_right` (int set\_steps)  
*Set the number of right motor steps.*

### Variables

- static int `left_speed` = 0
- static int `right_speed` = 0
- static int `nbr_pas_left` = 0
- static int `nbr_pas_right` = 0

### 7.75.1 Detailed Description

Manage the motors (with timer 4 and 5).

This module manage the motors with two timers: timer4 (motor left) and timer5 (motor right).

#### Warning

You can't use this module to control the motors if you are using the camera, because the camera's module also use timer4 and timer5.

A little exemple for the motors (e-puck turn on himself)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/e_motors.h>

int main(void)
{
    e_init_motors();
    e_set_speed_left(500); //go forward on half speed
    e_set_speed_right(-500); //go backward on half speed
    while(1) {}
}
```

#### Author

Code: Michael Bonani, Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

### 7.75.2 Function Documentation

7.75.2.1 void \_\_attribute\_\_((interrupt, auto\_psv, shadow))

7.75.2.2 int e\_get\_steps\_left ( void )

Give the number of left motor steps.

#### Returns

The number of phases steps made since the left motor is running.

7.75.2.3 int e\_get\_steps\_right ( void )

Give the number of right motor steps.

#### Returns

The number of phases steps made since the right motor is running.

## 7.75.2.4 void e\_init\_motors ( void )

Initialize the motors's ports.

Initialize the motors's agendas.

This function initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

See also

[e\\_init\\_port](#)

## 7.75.2.5 void e\_set\_speed\_left ( int motor\_speed )

Manage the left speed.

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controled by the timer5.

Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## 7.75.2.6 void e\_set\_speed\_right ( int motor\_speed )

Manage the right speed.

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer4.

Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## 7.75.2.7 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

### 7.75.2.8 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

#### Parameters

<code>set_steps</code>	The number of changed phases that you want set.
------------------------	---

## 7.75.3 Variable Documentation

7.75.3.1 int left\_speed = 0 [static]

7.75.3.2 int nbr\_pas\_left = 0 [static]

7.75.3.3 int nbr\_pas\_right = 0 [static]

7.75.3.4 int right\_speed = 0 [static]

## 7.76 motor\_led/advance\_one\_timer/e\_motors.h File Reference

Manage the motors (with timer2)

### Functions

- void [e\\_init\\_motors](#) (void)  
*Initialize the motors's agendas.*
- void [e\\_set\\_speed\\_left](#) (int motor\_speed)  
*Manage the left motor speed.*
- void [e\\_set\\_speed\\_right](#) (int motor\_speed)  
*Manage the right motor speed.*
- void [e\\_set\\_speed](#) (int linear\_speed, int angular\_speed)  
*Manage linear/angular speed.*
- void [e\\_set\\_steps\\_left](#) (int steps\_left)  
*Set the number of left motor steps.*
- void [e\\_set\\_steps\\_right](#) (int steps\_right)  
*Set the number of right motor steps.*
- int [e\\_get\\_steps\\_left](#) ()  
*Give the number of left motor steps.*
- int [e\\_get\\_steps\\_right](#) ()  
*Give the number of right motor steps.*



## 7.76.1 Detailed Description

Manage the motors (with timer2)

This module manage the motors with the agenda solution (timer2).

A little exemple to use the motors with agenda (e-puck turn on himself)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_motors.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    e_init_port ();
    e_init_motors ();
    e_set_speed(-500, 500);
    e_start_agendas_processing ();
    while(1) {}
}
```

See also

[e\\_agenda.h](#)

Author

Code: Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

## 7.76.2 Function Documentation

### 7.76.2.1 int e\_get\_steps\_left ( void )

Give the number of left motor steps.

Returns

The number of phases steps made since the left motor is running.

### 7.76.2.2 int e\_get\_steps\_right ( void )

Give the number of right motor steps.

Returns

The number of phases steps made since the right motor is running.

### 7.76.2.3 void e\_init\_motors ( void )

Initialize the motors's agendas.

This function initialize the agendas used by the motors. In fact it call [e\\_activate\\_agenda](#)(void (\*func)(void), int cycle) function.

See also

[e\\_activate\\_agenda](#)

This function initialize the agendas used by the motors. In fact it call "e\_activate\_agenda(void (\*func)(void), int cycle)" function.

See also

[e\\_activate\\_agenda](#)

Initialize the motors's agendas.

This function initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

See also

[e\\_init\\_port](#)

Initialize the motors's agendas.

This function configure timer3 and initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

See also

[e\\_init\\_port](#)

### 7.76.2.4 void e\_set\_speed ( int *linear\_speed*, int *angular\_speed* )

Manage linear/angular speed.

This function manage the speed of the motors according to the desired linear and angular speed.

Parameters

<i>linear_speed</i>	the speed in the axis of e-puck
<i>angular_speed</i>	the rotation speed (trigonometric)

### 7.76.2.5 void e\_set\_speed\_left ( int *motor\_speed* )

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### See also

[e\\_set\\_agenda\\_cycle](#)

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### See also

[e\\_set\\_agenda\\_cycle](#)

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controled by the timer5.

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controled by the timer3.

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### 7.76.2.6 void e\_set\_speed\_right ( int motor\_speed )

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer4.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer3.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

### 7.76.2.7 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

### 7.76.2.8 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

#### Parameters

<code>set_steps</code>	The number of changed phases that you want set.
------------------------	---

## 7.77 motor\_led/advance\_one\_timer/fast\_agenda/e\_motors.h File Reference

Manage the motors (with timer1, 2, 3)

### Functions

- void [e\\_init\\_motors](#) (void)  
*Initialize the motors's agendas.*
- void [e\\_set\\_speed\\_left](#) (int motor\_speed)  
*Manage the left motor speed.*
- void [e\\_set\\_speed\\_right](#) (int motor\_speed)  
*Manage the right motor speed.*
- void [e\\_set\\_speed](#) (int linear\_speed, int angular\_speed)  
*Manage linear/angular speed.*
- void [e\\_set\\_steps\\_left](#) (int steps\_left)  
*Set the number of left motor steps.*
- void [e\\_set\\_steps\\_right](#) (int steps\_right)  
*Set the number of right motor steps.*
- int [e\\_get\\_steps\\_left](#) ()  
*Give the number of left motor steps.*
- int [e\\_get\\_steps\\_right](#) ()  
*Give the number of right motor steps.*

### 7.77.1 Detailed Description

Manage the motors (with timer1, 2, 3)

This module manage the motors with the fast agenda solution (timer1, 2, 3).

#### See also

[e\\_agenda.h](#)

#### Author

Code: Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

## 7.77.2 Function Documentation

### 7.77.2.1 `int e_get_steps_left ( void )`

Give the number of left motor steps.

#### Returns

The number of phases steps made since the left motor is running.

### 7.77.2.2 `int e_get_steps_right ( void )`

Give the number of right motor steps.

#### Returns

The number of phases steps made since the right motor is running.

### 7.77.2.3 `void e_init_motors ( void )`

Initialize the motors's agendas.

This function initialize the agendas used by the motors. In fact it call [e\\_activate\\_agenda](#)(void (\*func)(void), int cycle) function.

#### See also

[e\\_activate\\_agenda](#)

This function initialize the agendas used by the motors. In fact it call "e\_activate\_agenda(void (\*func)(void), int cycle)" function.

#### See also

[e\\_activate\\_agenda](#)

Initialize the motors's agendas.

This function initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

#### See also

[e\\_init\\_port](#)

Initialize the motors's agendas.

This function configure timer3 and initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

#### See also

[e\\_init\\_port](#)

### 7.77.2.4 `void e_set_speed ( int linear_speed, int angular_speed )`

Manage linear/angular speed.

This function manage the speed of the motors according to the desired linear and angular speed.

## Parameters

<i>linear_speed</i>	the speed in the axis of e-puck
<i>angular_speed</i>	the rotation speed (trigonometric)

7.77.2.5 void e\_set\_speed\_left ( int *motor\_speed* )

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the timer5.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the timer3.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## 7.77.2.6 void e\_set\_speed\_right ( int motor\_speed )

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer4.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer3.



## Parameters

<code>motor_speed</code>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------------	---

## 7.77.2.7 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

## Parameters

<code>set_steps</code>	The number of changed phases that you want set.
------------------------	---

## 7.77.2.8 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

## Parameters

<code>set_steps</code>	The number of changed phases that you want set.
------------------------	---

## 7.78 motor\_led/e\_motors.h File Reference

Manage the motors (with timer 4 and 5).

## Functions

- void [e\\_init\\_motors](#) (void)  
*Initialize the motors's agendas.*
- void [e\\_set\\_speed\\_left](#) (int motor\_speed)  
*Manage the left motor speed.*
- void [e\\_set\\_speed\\_right](#) (int motor\_speed)  
*Manage the right motor speed.*
- int [e\\_get\\_steps\\_left](#) (void)  
*Give the number of left motor steps.*
- int [e\\_get\\_steps\\_right](#) (void)  
*Give the number of right motor steps.*
- void [e\\_set\\_steps\\_left](#) (int set\_steps)  
*Set the number of left motor steps.*
- void [e\\_set\\_steps\\_right](#) (int set\_steps)  
*Set the number of right motor steps.*

## 7.78.1 Detailed Description

Manage the motors (with timer 4 and 5).

This module manage the motors with two timers: timer4 (motor left) and timer5 (motor right).

### Warning

You can't use this module to control the motors if you are using the camera, because the camera's module also use timer4 and timer5.

A little exemple for the motors (e-puck turn on himself)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/e_motors.h>

int main(void)
{
    e_init_motors();
    e_set_speed_left(500); //go forward on half speed
    e_set_speed_right(-500); //go backward on half speed
    while(1) {}
}
```

### Author

Code: Michael Bonani, Francesco Mondada, Lucas Meier  
Doc: Jonathan Besuchet

## 7.78.2 Function Documentation

### 7.78.2.1 int e\_get\_steps\_left ( void )

Give the number of left motor steps.

#### Returns

The number of phases steps made since the left motor is running.

### 7.78.2.2 int e\_get\_steps\_right ( void )

Give the number of right motor steps.

#### Returns

The number of phases steps made since the right motor is running.

### 7.78.2.3 void e\_init\_motors ( void )

Initialize the motors's agendas.

This function initialize the agendas used by the motors. In fact it call [e\\_activate\\_agenda](#)(void (\*func)(void), int cycle) function.

See also

[e\\_activate\\_agenda](#)

This function initialize the agendas used by the motors. In fact it call "e\_activate\_agenda(void (\*func)(void), int cycle)" function.

See also

[e\\_activate\\_agenda](#)

Initialize the motors's agendas.

This function initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

See also

[e\\_init\\_port](#)

Initialize the motors's agendas.

This function configure timer3 and initialize the ports used by the motors. In fact it call "the e\_init\_port()" function.

See also

[e\\_init\\_port](#)

### 7.78.2.4 void e\_set\_speed\_left ( int motor\_speed )

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

See also

[e\\_set\\_agenda\\_cycle](#)

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### See also

[e\\_set\\_agenda\\_cycle](#)

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the timer5.

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the timer3.

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### 7.78.2.5 void e\_set\_speed\_right ( int motor\_speed )

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function [e\\_set\\_agenda\\_cycle](#)(void (\*func)(void), int cycle)).

#### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

#### See also

[e\\_set\\_agenda\\_cycle](#)

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controlled by the agenda (throw the function "e\_set\_agenda\_cycle(void (\*func)(void), int cycle)").

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## See also

[e\\_set\\_agenda\\_cycle](#)

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer4.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controled by the timer3.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## 7.78.2.6 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

## 7.78.2.7 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

## 7.79 motor\_led/advance\_one\_timer/e\_remote\_control.c File Reference

Manage the IR receiver module (timer2)

```
#include "e_remote_control.h"
#include "../e_epuck_ports.h"
#include "e_agenda.h"
#include "e_led.h"
```

### Functions

- void [e\\_init\\_remote\\_control](#) (void)  
*Initialise the IR receiver ports.*
- void [\\_\\_attribute\\_\\_](#) ((\_\_interrupt\_\_, auto\_psv))
- void [e\\_read\\_remote\\_control](#) (void)  
*Read the signal and stock the information.*
- unsigned char [e\\_get\\_check](#) (void)  
*Read the check bit.*
- unsigned char [e\\_get\\_address](#) (void)  
*Read the adress of the commande.*
- unsigned char [e\\_get\\_data](#) (void)  
*Read the data of the command.*

### Variables

- static unsigned char [address\\_temp](#) = 0
- static unsigned char [data\\_temp](#) = 0
- static unsigned char [check\\_temp](#) = 0
- static unsigned char [address](#) = 0
- static unsigned char [data](#) = 0
- static unsigned char [check](#) = 2

### 7.79.1 Detailed Description

Manage the IR receiver module (timer2)

This module manage the IR receiver with the agenda solution (timer2).

A little exemple to manage the IR remote (the body LED change his state when you press a button of the IR controller).

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_remote_control.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    int ir_check;
    int previous_check = 0;
    e_init_port();
    e_init_remote_control();
    e_start_agendas_processing();
    while(1)
    {
        ir_check = e_get_check();
        if(ir_check != previous_check)
            BODY_LED = BODY_LED^1;
        previous_check = ir_check;
    }
}
```

See also

[e\\_agenda.h](#)

Author

Code: Francesco Mondada, Lucas Meier

Doc: Jonathan Besuchet

## 7.79.2 Function Documentation

7.79.2.1 `void __attribute__((__interrupt__, auto_psv))`

7.79.2.2 `unsigned char e_get_address ( void )`

Read the address of the commande.

Returns

adress adress part of the signal

7.79.2.3 `unsigned char e_get_check ( void )`

Read the check bit.

Returns

check check bit of the signal

7.79.2.4 `unsigned char e_get_data ( void )`

Read the data of the command.

Returns

data data part of the signal

7.79.2.5 `void e_init_remote_control ( void )`

Initialise the IR receiver ports.

7.79.2.6 `void e_read_remote_control ( void )`

Read the signal and stock the information.

### 7.79.3 Variable Documentation

7.79.3.1 unsigned char address = 0 [static]

7.79.3.2 unsigned char address\_temp = 0 [static]

7.79.3.3 unsigned char check = 2 [static]

7.79.3.4 unsigned char check\_temp = 0 [static]

7.79.3.5 unsigned char data = 0 [static]

7.79.3.6 unsigned char data\_temp = 0 [static]

## 7.80 motor\_led/advance\_one\_timer/e\_remote\_control.h File Reference

Manage the LEDs with blinking possibility (timer2).

### Macros

- #define [BOTTOMR](#) 10
- #define [BOTTOMI](#) 11
- #define [STANDBY](#) 12
- #define [MUTE](#) 13
- #define [VOL\\_UP](#) 16
- #define [VOL\\_DOWN](#) 17
- #define [CHAN\\_UP](#) 32
- #define [CHAN\\_DOWN](#) 33
- #define [I\\_II](#) 35
- #define [OUT\\_AUX\\_1](#) 56

### Functions

- void [e\\_init\\_remote\\_control](#) (void)  
*Initialise the IR receiver ports.*
- void [e\\_read\\_remote\\_control](#) (void)  
*Read the signal and stock the information.*
- unsigned char [e\\_get\\_check](#) (void)  
*Read the check bit.*
- unsigned char [e\\_get\\_address](#) (void)  
*Read the adress of the commande.*
- unsigned char [e\\_get\\_data](#) (void)  
*Read the data of the command.*



## 7.80.1 Detailed Description

Manage the LEDs with blinking possibility (timer2).

Manage the IR receiver module (timer2)

Here we use the agenda solution to make the LED blinking.

A little exemple for LEDs blinking with agenda (all LEDs blink with 100ms delay)

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_led.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    e_init_port();
    e_activate_agenda(e_blink_led, 1000); //blink with 100ms
    e_start_agendas_processing();
    while(1) {}
}
```

See also

[e\\_agenda.h](#)

Author

Code: Valentin Longchamp

Doc: Jonathan Besuchet

This module manage the IR receiver with the agenda solution (timer2).

A little exemple to manage the IR remote (the body LED change his state when you press a button of the IR controller).

```
#include <p30F6014A.h>
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <motor_led/advance_one_timer/e_remote_control.h>
#include <motor_led/advance_one_timer/e_agenda.h>

int main(void)
{
    int ir_check;
    int previous_check = 0;
    e_init_port();
    e_init_remote_control();
    e_start_agendas_processing();
    while(1)
    {
        ir_check = e_get_check();
        if(ir_check != previous_check)
            BODY_LED = BODY_LED^1;
        previous_check = ir_check;
    }
}
```

See also

[e\\_agenda.h](#)

Author

Jonathan Besuchet

## 7.80.2 Macro Definition Documentation

7.80.2.1 `#define BOTTOMI 11`

7.80.2.2 `#define BOTTOMR 10`

7.80.2.3 `#define CHAN_DOWN 33`

7.80.2.4 `#define CHAN_UP 32`

7.80.2.5 `#define I_II 35`

7.80.2.6 `#define MUTE 13`

7.80.2.7 `#define OUT_AUX_1 56`

7.80.2.8 `#define STANDBY 12`

7.80.2.9 `#define VOL_DOWN 17`

7.80.2.10 `#define VOL_UP 16`

## 7.80.3 Function Documentation

7.80.3.1 `unsigned char e_get_address ( void )`

Read the address of the commande.

### Returns

address address part of the signal

7.80.3.2 `unsigned char e_get_check ( void )`

Read the check bit.

### Returns

check check bit of the signal

7.80.3.3 `unsigned char e_get_data ( void )`

Read the data of the command.

### Returns

data data part of the signal

## 7.80.3.4 void e\_init\_remote\_control ( void )

Initialise the IR receiver ports.

## 7.80.3.5 void e\_read\_remote\_control ( void )

Read the signal and stock the information.

## 7.81 motor\_led/advance\_one\_timer/fast\_agenda/e\_agenda\_fast.c File Reference

Manage the fast agendas (timer1, 2, 3).

```
#include "e_agenda_fast.h"
#include "../..//e_epuck_ports.h"
#include <stdlib.h>
```

## Macros

- #define EXIT\_OK 1

## Functions

- unsigned compute\_gcd (unsigned u, unsigned v)
- unsigned my\_ceil (float a)
- void e\_set\_timer\_speed (int timer, unsigned speed)
- int search\_best\_fit (unsigned cycle)
- void recompute\_speeds ()
- Agenda \* find\_function (void(\*func)(void))
- unsigned assign\_agenda (Agenda \*agenda)
- void migrate (int timer)
- void e\_start\_agendas\_processing (void)
  - *Initialise the accounting structure.*
- void e\_configure\_timer (int timer)
  - *Configure the timer(s) used.*
- void e\_start\_timer\_processing (int timer)
  - *Start the timer(s) used.*
- void e\_end\_agendas\_processing (int timer)
  - *Stop an agenda running.*
- void e\_activate\_agenda (void(\*func)(void), unsigned cycle)
  - *Activate a fast agenda.*
- void e\_activate\_motors (void(\*func1)(void), void(\*func2)(void))
- int e\_set\_motor\_speed (void(\*func)(void), unsigned cycle)
- int e\_destroy\_agenda (void(\*func)(void))
  - *Destroy an agenda.*
- int e\_set\_agenda\_cycle (void(\*func)(void), unsigned cycle)
- int e\_reset\_agenda (void(\*func)(void))
  - *Reset an agenda.*
- int e\_pause\_agenda (void(\*func)(void))
  - *Pause an agenda.*
- int e\_restart\_agenda (void(\*func)(void))
  - *Restart an agenda previously paused.*
- void \_\_attribute\_\_ ((\_\_interrupt\_\_, auto\_psv))
  - *Interrupt from timer1.*

## Variables

- static struct [AgendaList agenda\\_list](#)

### 7.81.1 Detailed Description

Manage the fast agendas (timer1, 2, 3).

This module manage the fast agendas with the timer1, 2, 3.

An agenda is a structure made to work as chained list. It contains: the function you want to launch, the time setup between two launching events, a counter to measure the current time, a pointer to the next element of the list.

Each times the timer1, 2, 3 has an interrupt, the corresponding agenda chained list is scanned to look if an agenda has to be treated according to the cycle value and current counter value.

If one (or more) agenda has to be treated, his callback function is launch.

#### Author

Code: Julien Hubert

Doc: Jonathan Besuchet

### 7.81.2 Macro Definition Documentation

7.81.2.1 `#define EXIT_OK 1`

### 7.81.3 Function Documentation

7.81.3.1 `void __attribute__ ( (__interrupt__, auto_psv) )`

Interrupt from timer1.

Interrupt from timer3.

Interrupt from timer2.

Parse the chained list of agenda.

Increment counter only.

Check if agenda has to be treated according to the cycle value and current counter value.

Do it for number of cycle positive or null.

Check if a service has to be activated.

7.81.3.2 `unsigned assign_agenda ( Agenda * agenda )`

7.81.3.3 `unsigned compute_gcd ( unsigned u, unsigned v )`

7.81.3.4 `void e_activate_agenda ( void (*)(void) func, unsigned cycle )`

Activate a fast agenda.

Activate an agenda and allocate memory for him if there isn't already an agenda with the same callback function (the agenda is active but isn't processed if he have a null cycle value).

The appropriate timer is automatically selected.

## Parameters

<i>func</i>	function called if the cycle value is reached by the counter
<i>cycle</i>	cycle value in millisec/10

7.81.3.5 void e\_activate\_motors ( void(\*)*(void) func1*, void(\*)*(void) func2* )

7.81.3.6 void e\_configure\_timer ( int *timer* )

Configure the timer(s) used.

Configure one or all the timers to be used by the agenda

## Parameters

<i>timer</i>	the timer's number to configure (between [0-3] with 0 configuring all of them)
--------------	--

7.81.3.7 int e\_destroy\_agenda ( void(\*)*(void) func* )

Destroy an agenda.

Destroy the agenda with a given callback function

## Parameters

<i>func</i>	function to test
-------------	------------------

## Returns

int return the success of the destruction (EXIT\_OK for successfull, AG\_NOT\_FOUND for unsuccessful).

7.81.3.8 void e\_end\_agendas\_processing ( int *timer* )

Stop an agenda running.

Stop the agendas running on one particular timer (the memory allocated for the agenda isn't freed, use e\_destroy\_agenda for that).

## Parameters

<i>timer</i>	the timer's number [0-3]. 0 stops all the timers.
--------------	---

## See also

[e\\_destroy\\_agenda](#)

7.81.3.9 `int e_pause_agenda ( void(*) (void) func )`

Pause an agenda.

Pause an agenda but do not reset its information.

Parameters

<i>func</i>	function to pause
-------------	-------------------

7.81.3.10 `int e_reset_agenda ( void(*) (void) func )`

Reset an agenda.

Reset an agenda's counter.

Reset an agenda's counter with a given callback function

Parameters

<i>func</i>	function to reset
-------------	-------------------

7.81.3.11 `int e_restart_agenda ( void(*) (void) func )`

Restart an agenda previously paused.

Restart an agenda previously paused

Parameters

<i>func</i>	function to restart
-------------	---------------------

7.81.3.12 `int e_set_agenda_cycle ( void(*) (void) func, unsigned cycle )`

7.81.3.13 `int e_set_motor_speed ( void(*) (void) func, unsigned cycle )`

7.81.3.14 `void e_set_timer_speed ( int timer, unsigned speed )`

7.81.3.15 `void e_start_agendas_processing ( void )`

Initialise the accounting structure.

Start the agendas processing.

Start the agendas processing by initialising the accounting structures.

Don't activate any timer which is done by `e_start_timer_processing`.

See also

[e\\_start\\_timer\\_processing](#)

7.81.3.16 void e\_start\_timer\_processing ( int *timer* )

Start the timer(s) used.

Activate one or all the timers to be used by the agenda.

Parameters

<i>timer</i>	the timer's number to activate (between [0-3] with 0 activating all of them)
--------------	--

7.81.3.17 Agenda\* find\_function ( void(\*) (void) *func* )

7.81.3.18 void migrate ( int *timer* )

7.81.3.19 unsigned my\_ceil ( float *a* )

7.81.3.20 void recompute\_speeds ( )

7.81.3.21 int search\_best\_fit ( unsigned *cycle* )

## 7.81.4 Variable Documentation

7.81.4.1 struct AgendaList agenda\_list [static]

## 7.82 motor\_led/advance\_one\_timer/fast\_agenda/e\_agenda\_fast.h File Reference

Manage the fast agendas (timer1, 2, 3).

### Data Structures

- struct [AgendaType](#)  
*struct Agenda as chained list*
- struct [AgendaList](#)  
*Manage the differents agendas lists.*

### Macros

- #define [AG\\_ALREADY\\_CREATED](#) 1
- #define [AG\\_NOT\\_FOUND](#) 2

### Typedefs

- typedef struct [AgendaType](#) [Agenda](#)

## Functions

- void `e_start_agendas_processing` (void)  
*Start the agendas processing.*
- void `e_start_timer_processing` (int timer)  
*Start the timer(s) used.*
- void `e_end_agendas_processing` (int timer)  
*Stop an agenda running.*
- void `e_configure_timer` (int timer)  
*Configure the timer(s) used.*
- void `e_activate_agenda` (void(\*func)(void), unsigned cycle)  
*Activate a fast agenda.*
- void `e_activate_motors` (void(\*func1)(void), void(\*func2)(void))
- int `e_set_motor_speed` (void(\*func)(void), unsigned cycle)
- int `e_destroy_agenda` (void(\*func)(void))  
*Destroy an agenda.*
- int `e_set_agenda_cycle` (void(\*func)(void), unsigned cycle)
- int `e_reset_agenda` (void(\*func)(void))  
*Reset an agenda's counter.*
- int `e_pause_agenda` (void(\*func)(void))  
*Pause an agenda.*
- int `e_restart_agenda` (void(\*func)(void))  
*Restart an agenda previously paused.*

### 7.82.1 Detailed Description

Manage the fast agendas (timer1, 2, 3).

This module manage the fast agendas with the timer1, 2, 3.

An agenda is a structure made to work as chained list. It contains: the function you want to launch, the time setup between two launching events, a counter to measure the current time, a pointer to the next element of the list.

Each times the timer1, 2, 3 has an interrupt, the corresponding agenda chained list is scanned to look if an agenda has to be treated according to the cycle value and current counter value.

If one (or more) agenda has to be treated, his callback function is launch.

#### Author

Code: Julien Hubert  
Doc: Jonathan Besuchet



## 7.82.2 Macro Definition Documentation

7.82.2.1 `#define AG_ALREADY_CREATED 1`

7.82.2.2 `#define AG_NOT_FOUND 2`

## 7.82.3 Typedef Documentation

7.82.3.1 `typedef struct AgendaType Agenda`

## 7.82.4 Function Documentation

7.82.4.1 `void e_activate_agenda ( void(*)(void) func, unsigned cycle )`

Activate a fast agenda.

Activate an agenda and allocate memory for him if there isn't already an agenda with the same callback function (the agenda is active but isn't processed if he have a null cycle value). The appropriate timer is automatically selected.

### Parameters

<i>func</i>	function called if the cycle value is reached by the counter
<i>cycle</i>	cycle value in millisec/10

7.82.4.2 `void e_activate_motors ( void(*)(void) func1, void(*)(void) func2 )`

7.82.4.3 `void e_configure_timer ( int timer )`

Configure the timer(s) used.

Configure one or all the timers to be used by the agenda

### Parameters

<i>timer</i>	the timer's number to configure (between [0-3] with 0 configuring all of them)
--------------	--

7.82.4.4 `int e_destroy_agenda ( void(*)(void) func )`

Destroy an agenda.

Destroy the agenda with a given callback function.

### Parameters

<i>func</i>	function to test
-------------	------------------

**Returns**

[EXIT\\_OK](#) if the agenda has been destroyed, [AG\\_NOT\\_FOUND](#) otherwise

Destroy the agenda with a given callback function

**Parameters**

<i>func</i>	function to test
-------------	------------------

**Returns**

int return the success of the destruction (EXIT\_OK for successfull, AG\_NOT\_FOUND for unsuccessful).

**7.82.4.5 void e\_end\_agendas\_processing ( int timer )**

Stop an agenda running.

Stop the agendas running on one particular timer (the memory allocated for the agenda isn't freed, use [e\\_destroy\\_agenda](#) for that).

**Parameters**

<i>timer</i>	the timer's number [0-3]. 0 stops all the timers.
--------------	---

**See also**

[e\\_destroy\\_agenda](#)

**7.82.4.6 int e\_pause\_agenda ( void(\*) (void) func )**

Pause an agenda.

Pause an agenda but do not reset its information.

**Parameters**

<i>func</i>	function to pause
-------------	-------------------

**Returns**

[EXIT\\_OK](#) the agenda has been paused, [AG\\_NOT\\_FOUND](#) otherwise

Pause an agenda but do not reset its information.

**Parameters**

<i>func</i>	function to pause
-------------	-------------------

#### 7.82.4.7 int e\_reset\_agenda ( void(\*) (void) func )

Reset an agenda's counter.

Reset an agenda's counter with a given callback function.

##### Parameters

<i>func</i>	function to reset
-------------	-------------------

##### Returns

[EXIT\\_OK](#) if the cycle of the agenda has been reseted, [AG\\_NOT\\_FOUND](#) otherwise

##### Warning

This function RESET the agenda, if you just want a pause tell [e\\_pause\\_agenda](#)(void (\*func)(void))

##### See also

[e\\_pause\\_agenda](#)

Reset an agenda's counter.

Reset an agenda's counter with a given callback function

##### Parameters

<i>func</i>	function to reset
-------------	-------------------

#### 7.82.4.8 int e\_restart\_agenda ( void(\*) (void) func )

Restart an agenda previously paused.

Restart an agenda previously paused.

##### Parameters

<i>func</i>	function to restart
-------------	---------------------

##### Returns

[EXIT\\_OK](#) if he agenda has been restarted, [AG\\_NOT\\_FOUND](#) otherwise

##### See also

[e\\_pause\\_agenda](#)

Restart an agenda previously paused

## Parameters

<i>func</i>	function to restart
-------------	---------------------

7.82.4.9 `int e_set_agenda_cycle ( void(*)(void) func, unsigned cycle )`

7.82.4.10 `int e_set_motor_speed ( void(*)(void) func, unsigned cycle )`

7.82.4.11 `void e_start_agendas_processing ( void )`

Start the agendas processing.

Start the agendas processing by starting the Timer2.

Start the agendas processing.

Start the agendas processing by initialising the accounting structures.  
Don't activate any timer which is done by `e_start_timer_processing`.

## See also

[e\\_start\\_timer\\_processing](#)

7.82.4.12 `void e_start_timer_processing ( int timer )`

Start the timer(s) used.

Activate one or all the timers to be used by the agenda.

## Parameters

<i>timer</i>	the timer's number to activate (between [0-3] with 0 activating all of them)
--------------	--

## 7.83 motor\_led/e\_epuck\_ports.h File Reference

Define all the usefull names corresponding of e-puck's hardware.

```
#include "p30F6014A.h"
```

### Macros

- `#define FOSC 7.3728e6`
- `#define PLL 8.0`
- `#define FCY ((FOSC*PLL)/(4.0))`

- #define `MILLISEC` (`FCY/1.0e3`)
- #define `MICROSEC` (`FCY/1.0e6`)
- #define `NANOSEC` (`FCY/1.0e9`)
- #define `TCY_PIC` (`1e9/FCY`)
- #define `INTERRUPT_DELAY` (`10*TCY_PIC`)
- #define `TRUE` 1
- #define `FALSE` 0
- #define `OUTPUT_PIN` 0
- #define `LED0_LATA6`
- #define `LED1_LATA7`
- #define `LED2_LATA9`
- #define `LED3_LATA12`
- #define `LED4_LATA10`
- #define `LED5_LATA13`
- #define `LED6_LATA14`
- #define `LED7_LATA15`
- #define `LED0_DIR_TRISA6`
- #define `LED1_DIR_TRISA7`
- #define `LED2_DIR_TRISA9`
- #define `LED3_DIR_TRISA12`
- #define `LED4_DIR_TRISA10`
- #define `LED5_DIR_TRISA13`
- #define `LED6_DIR_TRISA14`
- #define `LED7_DIR_TRISA15`
- #define `FRONT_LED_LATC1`
- #define `FRONT_LED_DIR_TRISC1`
- #define `BODY_LED_LATC2`
- #define `BODY_LED_DIR_TRISC2`
- #define `PULSE_IR0_LATF7`
- #define `PULSE_IR1_LATF8`
- #define `PULSE_IR2_LATG0`
- #define `PULSE_IR3_LATG1`
- #define `PULSE_IR0_DIR_TRISF7`
- #define `PULSE_IR1_DIR_TRISF8`
- #define `PULSE_IR2_DIR_TRISG0`
- #define `PULSE_IR3_DIR_TRISG1`
- #define `IR0` 8
- #define `IR1` 9
- #define `IR2` 10
- #define `IR3` 11
- #define `IR4` 12
- #define `IR5` 13
- #define `IR6` 14
- #define `IR7` 15
- #define `MIC1` 2
- #define `MIC2` 3
- #define `MIC3` 4
- #define `ACCX` 5
- #define `ACCY` 6
- #define `ACCZ` 7
- #define `AUDIO_ON_LATF0`
- #define `AUDIO_ON_DIR_TRISF0`
- #define `MOTOR1_PHA_LATD0`
- #define `MOTOR1_PHB_LATD1`
- #define `MOTOR1_PHC_LATD2`

- #define MOTOR1\_PHD\_LATD3
- #define MOTOR2\_PHA\_LATD4
- #define MOTOR2\_PHB\_LATD5
- #define MOTOR2\_PHC\_LATD6
- #define MOTOR2\_PHD\_LATD7
- #define MOTOR1\_PHA\_DIR\_TRISD0
- #define MOTOR1\_PHB\_DIR\_TRISD1
- #define MOTOR1\_PHC\_DIR\_TRISD2
- #define MOTOR1\_PHD\_DIR\_TRISD3
- #define MOTOR2\_PHA\_DIR\_TRISD4
- #define MOTOR2\_PHB\_DIR\_TRISD5
- #define MOTOR2\_PHC\_DIR\_TRISD6
- #define MOTOR2\_PHD\_DIR\_TRISD7
- #define CAM\_RESET\_LATC13
- #define CAM\_RESET\_DIR\_TRISC13
- #define SIO\_D\_LATG3
- #define SIO\_D\_DIR\_TRISG3
- #define SIO\_C\_LATG2
- #define SIO\_C\_DIR\_TRISG2
- #define INPUT\_PIN 1
- #define BATT\_LOW\_RF1
- #define BATT\_LOW\_DIR\_TRISF1
- #define SELECTOR0\_RG6
- #define SELECTOR1\_RG7
- #define SELECTOR2\_RG8
- #define SELECTOR3\_RG9
- #define SELECTOR0\_DIR\_TRISG6
- #define SELECTOR1\_DIR\_TRISG7
- #define SELECTOR2\_DIR\_TRISG8
- #define SELECTOR3\_DIR\_TRISG9
- #define REMOTE\_RF6
- #define REMOTE\_DIR\_TRISF6
- #define CAM\_DATA PORTD;
- #define CAM\_y0\_RD8
- #define CAM\_y1\_RD9
- #define CAM\_y2\_RD10
- #define CAM\_y3\_RD11
- #define CAM\_y4\_RD12
- #define CAM\_y5\_RD13
- #define CAM\_y6\_RD14
- #define CAM\_y7\_RD15
- #define CAM\_y0\_DIR\_TRISD8
- #define CAM\_y1\_DIR\_TRISD9
- #define CAM\_y2\_DIR\_TRISD10
- #define CAM\_y3\_DIR\_TRISD11
- #define CAM\_y4\_DIR\_TRISD12
- #define CAM\_y5\_DIR\_TRISD13
- #define CAM\_y6\_DIR\_TRISD14
- #define CAM\_y7\_DIR\_TRISD15
- #define CAM\_PWDN\_RC2
- #define CAM\_VSYNC\_RC4
- #define CAM\_HREF\_RC3
- #define CAM\_PCLK\_RC14
- #define CAM\_PWDN\_DIR\_TRISC2
- #define CAM\_VSYNC\_DIR\_TRISC4

- #define `CAM_HREF_DIR_TRISC3`
- #define `CAM_PCLK_DIR_TRISC14`
- #define `NOP()` {\_\_asm\_\_ volatile ("nop");}
- #define `CLRWDT()` {\_\_asm\_\_ volatile ("clrwdt");}
- #define `SLEEP()` {\_\_asm\_\_ volatile ("pwrsav #0");}
- #define `IDLE()` {\_\_asm\_\_ volatile ("pwrsav #1");}
- #define `INTERRUPT_OFF()` {\_\_asm\_\_ volatile ("disi #10000");}
- #define `INTERRUPT_ON()` {\_\_asm\_\_ volatile ("disi #2");}
- #define `RESET()` {\_\_asm\_\_ volatile ("reset");}
- #define `STOP_TMR1` IEC0bits.T11E = 0
- #define `STOP_TMR2` IEC0bits.T21E = 0
- #define `STOP_TMR3` IEC0bits.T31E = 0
- #define `STOP_TMR4` IEC1bits.T41E = 0
- #define `STOP_TMR5` IEC1bits.T51E = 0

### 7.83.1 Detailed Description

Define all the usefull names corresponding of e-puck's hardware.

#### Author

Code: Michael Bonani, Francesco Mondada, Davis Dadie  
Doc: Jonathan Besuchet

### 7.83.2 Macro Definition Documentation

7.83.2.1 #define `ACCX` 5

7.83.2.2 #define `ACCY` 6

7.83.2.3 #define `ACCZ` 7

7.83.2.4 #define `AUDIO_ON_LATF0`

7.83.2.5 #define `AUDIO_ON_DIR_TRISF0`

7.83.2.6 #define `BATT_LOW_RF1`

7.83.2.7 #define `BATT_LOW_DIR_TRISF1`

7.83.2.8 #define `BODY_LED_LATC2`

7.83.2.9 #define `BODY_LED_DIR_TRISC2`

7.83.2.10 #define `CAM_DATA` PORTD;

7.83.2.11 #define `CAM_HREF_RC3`

7.83.2.12 #define CAM\_HREF\_DIR\_TRISC3

7.83.2.13 #define CAM\_PCLK\_RC14

7.83.2.14 #define CAM\_PCLK\_DIR\_TRISC14

7.83.2.15 #define CAM\_PWDN\_RC2

7.83.2.16 #define CAM\_PWDN\_DIR\_TRISC2

7.83.2.17 #define CAM\_RESET\_LATC13

7.83.2.18 #define CAM\_RESET\_DIR\_TRISC13

7.83.2.19 #define CAM\_VSYNC\_RC4

7.83.2.20 #define CAM\_VSYNC\_DIR\_TRISC4

7.83.2.21 #define CAM\_y0\_RD8

7.83.2.22 #define CAM\_y0\_DIR\_TRISD8

7.83.2.23 #define CAM\_y1\_RD9

7.83.2.24 #define CAM\_y1\_DIR\_TRISD9

7.83.2.25 #define CAM\_y2\_RD10

7.83.2.26 #define CAM\_y2\_DIR\_TRISD10

7.83.2.27 #define CAM\_y3\_RD11

7.83.2.28 #define CAM\_y3\_DIR\_TRISD11

7.83.2.29 #define CAM\_y4\_RD12

7.83.2.30 #define CAM\_y4\_DIR\_TRISD12

7.83.2.31 #define CAM\_y5\_RD13

7.83.2.32 #define CAM\_y5\_DIR\_TRISD13

7.83.2.33 #define CAM\_y6\_RD14

7.83.2.34 #define CAM\_y6\_DIR\_TRISD14



- 7.83.2.35 `#define CAM_y7_RD15`
- 7.83.2.36 `#define CAM_y7_DIR_TRISD15`
- 7.83.2.37 `#define CLRWDT( ) {__asm__ volatile ("clrwdt");}`
- 7.83.2.38 `#define FALSE 0`
- 7.83.2.39 `#define FCY ((FOSC*PLL)/(4.0))`
- 7.83.2.40 `#define FOSC 7.3728e6`
- 7.83.2.41 `#define FRONT_LED_LATC1`
- 7.83.2.42 `#define FRONT_LED_DIR_TRISC1`
- 7.83.2.43 `#define IDLE( ) {__asm__ volatile ("pwrsav #1");}`
- 7.83.2.44 `#define INPUT_PIN 1`
- 7.83.2.45 `#define INTERRUPT_DELAY (10*TCY_PIC)`
- 7.83.2.46 `#define INTERRUPT_OFF( ) {__asm__ volatile ("disi #10000");}`
- 7.83.2.47 `#define INTERRUPT_ON( ) {__asm__ volatile ("disi #2");}`
- 7.83.2.48 `#define IR0 8`
- 7.83.2.49 `#define IR1 9`
- 7.83.2.50 `#define IR2 10`
- 7.83.2.51 `#define IR3 11`
- 7.83.2.52 `#define IR4 12`
- 7.83.2.53 `#define IR5 13`
- 7.83.2.54 `#define IR6 14`
- 7.83.2.55 `#define IR7 15`
- 7.83.2.56 `#define LED0_LATA6`
- 7.83.2.57 `#define LED0_DIR_TRISA6`

7.83.2.58 `#define LED1_LATA7`

7.83.2.59 `#define LED1_DIR_TRISA7`

7.83.2.60 `#define LED2_LATA9`

7.83.2.61 `#define LED2_DIR_TRISA9`

7.83.2.62 `#define LED3_LATA12`

7.83.2.63 `#define LED3_DIR_TRISA12`

7.83.2.64 `#define LED4_LATA10`

7.83.2.65 `#define LED4_DIR_TRISA10`

7.83.2.66 `#define LED5_LATA13`

7.83.2.67 `#define LED5_DIR_TRISA13`

7.83.2.68 `#define LED6_LATA14`

7.83.2.69 `#define LED6_DIR_TRISA14`

7.83.2.70 `#define LED7_LATA15`

7.83.2.71 `#define LED7_DIR_TRISA15`

7.83.2.72 `#define MIC1 2`

7.83.2.73 `#define MIC2 3`

7.83.2.74 `#define MIC3 4`

7.83.2.75 `#define MICROSEC (FCY/1.0e6)`

7.83.2.76 `#define MILLISEC (FCY/1.0e3)`

7.83.2.77 `#define MOTOR1_PHA_LATD0`

7.83.2.78 `#define MOTOR1_PHA_DIR_TRISD0`

7.83.2.79 `#define MOTOR1_PHB_LATD1`

7.83.2.80 `#define MOTOR1_PHB_DIR_TRISD1`

- 7.83.2.81 `#define MOTOR1_PHC_LATD2`
- 7.83.2.82 `#define MOTOR1_PHC_DIR_TRISD2`
- 7.83.2.83 `#define MOTOR1_PHD_LATD3`
- 7.83.2.84 `#define MOTOR1_PHD_DIR_TRISD3`
- 7.83.2.85 `#define MOTOR2_PHA_LATD4`
- 7.83.2.86 `#define MOTOR2_PHA_DIR_TRISD4`
- 7.83.2.87 `#define MOTOR2_PHB_LATD5`
- 7.83.2.88 `#define MOTOR2_PHB_DIR_TRISD5`
- 7.83.2.89 `#define MOTOR2_PHC_LATD6`
- 7.83.2.90 `#define MOTOR2_PHC_DIR_TRISD6`
- 7.83.2.91 `#define MOTOR2_PHD_LATD7`
- 7.83.2.92 `#define MOTOR2_PHD_DIR_TRISD7`
- 7.83.2.93 `#define NANOSEC (FCY/1.0e9)`
- 7.83.2.94 `#define NOP( ) {__asm__ volatile ("nop");}`
- 7.83.2.95 `#define OUTPUT_PIN 0`
- 7.83.2.96 `#define PLL 8.0`
- 7.83.2.97 `#define PULSE_IR0_LATF7`
- 7.83.2.98 `#define PULSE_IR0_DIR_TRISF7`
- 7.83.2.99 `#define PULSE_IR1_LATF8`
- 7.83.2.100 `#define PULSE_IR1_DIR_TRISF8`
- 7.83.2.101 `#define PULSE_IR2_LATG0`
- 7.83.2.102 `#define PULSE_IR2_DIR_TRISG0`
- 7.83.2.103 `#define PULSE_IR3_LATG1`

```
7.83.2.104 #define PULSE_IR3_DIR_TRISG1
7.83.2.105 #define REMOTE_RF6
7.83.2.106 #define REMOTE_DIR_TRISF6
7.83.2.107 #define RESET( ) {__asm__ volatile ("reset");}
7.83.2.108 #define SELECTOR0_RG6
7.83.2.109 #define SELECTOR0_DIR_TRISG6
7.83.2.110 #define SELECTOR1_RG7
7.83.2.111 #define SELECTOR1_DIR_TRISG7
7.83.2.112 #define SELECTOR2_RG8
7.83.2.113 #define SELECTOR2_DIR_TRISG8
7.83.2.114 #define SELECTOR3_RG9
7.83.2.115 #define SELECTOR3_DIR_TRISG9
7.83.2.116 #define SIO_C_LATG2
7.83.2.117 #define SIO_C_DIR_TRISG2
7.83.2.118 #define SIO_D_LATG3
7.83.2.119 #define SIO_D_DIR_TRISG3
7.83.2.120 #define SLEEP( ) {__asm__ volatile ("pwrsav #0");}
7.83.2.121 #define STOP_TMR1 IEC0bits.T1IE = 0
7.83.2.122 #define STOP_TMR2 IEC0bits.T2IE = 0
7.83.2.123 #define STOP_TMR3 IEC0bits.T3IE = 0
7.83.2.124 #define STOP_TMR4 IEC1bits.T4IE = 0
7.83.2.125 #define STOP_TMR5 IEC1bits.T5IE = 0
7.83.2.126 #define TCY_PIC (1e9/FCY)
7.83.2.127 #define TRUE 1
```

## 7.84 motor\_led/e\_init\_port.c File Reference

Initialize the ports on standard configuration.

```
#include "e_epuck_ports.h"
#include "../I2C/e_I2C_protocol.h"
#include "../acc_gyro/e_lsm330.h"
```

## Functions

- `_FOSC` (CSW\_FSCM\_OFF & XT\_PLL8)
- `_FWDT` (WDT\_OFF)
- `_FBORPOR` (PBOR\_OFF & MCLR\_EN)
- `_FGS` (CODE\_PROT\_OFF)
- void `testAccGyroPresence` ()
- void `e_init_port` (void)  
*Initialize all ports (in/out)*
- unsigned char `isEpuckVersion1_3` (void)

## Variables

- unsigned char `isPresentFlag` = 0

### 7.84.1 Detailed Description

Initialize the ports on standard configuration.

#### Author

Code: Michael Bonani, Francesco Mondada, Davis Dadie  
Doc: Jonathan Besuchet

### 7.84.2 Function Documentation

7.84.2.1 `_FBORPOR` ( `PBOR_OFF` & `MCLR_EN` )

7.84.2.2 `_FGS` ( `CODE_PROT_OFF` )

7.84.2.3 `_FOSC` ( `CSW_FSCM_OFF` & `XT_PLL8` )

7.84.2.4 `_FWDT` ( `WDT_OFF` )

7.84.2.5 void `e_init_port` ( void )

Initialize all ports (in/out)

Call this method to set all the standards output components (LEDs, IR, camera, motors, I2C, audio) on their defaults values and set their corresponding PIN to "output". The method also set the corresponding PIN to "input" for all the standards inputs components (IR receiver, selector, camera, battery level).

7.84.2.6 unsigned char isEpuckVersion1\_3 ( void )

7.84.2.7 void testAccGyroPresence ( )

### 7.84.3 Variable Documentation

7.84.3.1 unsigned char isPresentFlag = 0

## 7.85 motor\_led/e\_init\_port.h File Reference

Initialize the ports on standard configuration.

### Functions

- void [e\\_init\\_port](#) (void)  
*Initialize all ports (in/out)*
- unsigned char [isEpuckVersion1\\_3](#) (void)

### 7.85.1 Detailed Description

Initialize the ports on standard configuration.

#### Author

Code: Michael Bonani, Francesco Mondada, Davis Dadie  
Doc: Jonathan Besuchet

### 7.85.2 Function Documentation

7.85.2.1 void [e\\_init\\_port](#) ( void )

Initialize all ports (in/out)

Call this method to set all the standards output components (LEDs, IR, camera, motors, I2C, audio) on their defaults values and set their corresponding PIN to "output". The method also set the corresponding PIN to "input" for all the standards inputs components (IR receiver, selector, camera, battery level).

7.85.2.2 unsigned char [isEpuckVersion1\\_3](#) ( void )

## 7.86 motor\_led/e\_motors\_timer3.c File Reference

Initialize the ports on standard configuration.

```
#include <stdlib.h>
#include "e_epuck_ports.h"
#include "e_init_port.h"
```

## Functions

- void `__attribute__` ((interrupt, auto\_psv, shadow))
- int `e_get_steps_left` (void)  
*Give the number of left motor steps.*
- void `e_set_steps_left` (int set\_steps)  
*Set the number of left motor steps.*
- int `e_get_steps_right` (void)  
*Give the number of right motor steps.*
- void `e_set_steps_right` (int set\_steps)  
*Set the number of right motor steps.*
- void `e_set_speed_left` (int motor\_speed)  
*Manage the left speed.*
- void `e_set_speed_right` (int motor\_speed)  
*Manage the right speed.*
- void `e_init_motors` (void)  
*Initialize the motors's ports.*

## Variables

- static int `left_speed` = 0
- static int `right_speed` = 0
- static int `nbr_pas_left` = 0
- static int `nbr_pas_right` = 0
- static int `motor_counter_left` =0
- static int `motor_counter_right` =0
- static int `motor_counter_left_init` =0
- static int `motor_counter_right_init` =0

### 7.86.1 Detailed Description

Initialize the ports on standard configuration.

Manage the motors (with timer3).

#### Author

Code: Michael Bonani, Francesco Mondada, Davis Dadie  
Doc: Jonathan Besuchet

This module manage the two motors with one timer: timer3.

#### Author

Code: Michael Bonani, Francesco Mondada, Lucas Meier, Xavier Raemy  
Doc: Jonathan Besuchet

## 7.86.2 Function Documentation

7.86.2.1 `void __attribute__ ( (interrupt, auto_psv, shadow) )`

7.86.2.2 `int e_get_steps_left ( void )`

Give the number of left motor steps.

### Returns

The number of phases steps made since the left motor is running.

7.86.2.3 `int e_get_steps_right ( void )`

Give the number of right motor steps.

### Returns

The number of phases steps made since the right motor is running.

7.86.2.4 `void e_init_motors ( void )`

Initialize the motors's ports.

Initialize the motors's agendas.

This function configure timer3 and initialize the ports used by the motors. In fact it call "the `e_init_port()`" function.

### See also

[e\\_init\\_port](#)

7.86.2.5 `void e_set_speed_left ( int motor_speed )`

Manage the left speed.

Manage the left motor speed.

This function manage the left motor speed by changing the MOTOR1 phases. The changing phases frequency (=> speed) is controlled by the timer3.

### Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---



## 7.86.2.6 void e\_set\_speed\_right ( int motor\_speed )

Manage the right speed.

Manage the right motor speed.

This function manage the right motor speed by changing the MOTOR2 phases. The changing phases frequency (=> speed) is controlled by the timer3.

## Parameters

<i>motor_speed</i>	from -1000 to 1000 give the motor speed in steps/s, positive value to go forward and negative to go backward.
--------------------	---

## 7.86.2.7 void e\_set\_steps\_left ( int set\_steps )

Set the number of left motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

## 7.86.2.8 void e\_set\_steps\_right ( int set\_steps )

Set the number of right motor steps.

## Parameters

<i>set_steps</i>	The number of changed phases that you want set.
------------------	---

## 7.86.3 Variable Documentation

7.86.3.1 int left\_speed = 0 [static]

7.86.3.2 int motor\_counter\_left =0 [static]

7.86.3.3 int motor\_counter\_left\_init =0 [static]

7.86.3.4 int motor\_counter\_right =0 [static]

7.86.3.5 int motor\_counter\_right\_init =0 [static]

7.86.3.6 int nbr\_pas\_left = 0 [static]

7.86.3.7 `int nbr_pas_right = 0` `[static]`

7.86.3.8 `int right_speed = 0` `[static]`

## 7.87 `uart/e_epuck_ports.inc` File Reference

## 7.88 `uart/e_uart_char.h` File Reference

Manage UART.

### Macros

- `#define BAUD115200` 7
- `#define BAUD230400` 3
- `#define BAUD460800` 1
- `#define BAUD921600` 0

### Functions

- void `e_init_uart1` (void)  
*Init uart 1 at 115200bps, 8 data bits, 1 stop bit, Enable ISR for RX and TX.*
- int `e_ischar_uart1` ()  
*Check if something is comming on uart 1.*
- int `e_getchar_uart1` (char \*car)  
*If available, read 1 char and put it in pointer.*
- void `e_send_uart1_char` (const char \*buff, int length)  
*Send a buffer of char of size length.*
- int `e_uart1_sending` (void)  
*To check if the sending operation is done.*
- void `e_init_uart2` (int)  
*Init uart 2 at 115200bps, 8 data bits, 1 stop bit, Enable ISR for RX and TX.*
- int `e_ischar_uart2` ()  
*Check if something is comming on uart 2.*
- int `e_getchar_uart2` (char \*car)  
*If available, read 1 char and put it in pointer.*
- void `e_send_uart2_char` (const char \*buff, int length)  
*Send a buffer of char of size length.*
- int `e_uart2_sending` (void)  
*To check if the sending operation is done.*

### Variables

- void \* `e_uart1_int_clr_addr`
- int `e_uart1_int_clr_mask`
- void \* `e_uart2_int_clr_addr`
- int `e_uart2_int_clr_mask`

## 7.88.1 Detailed Description

Manage UART.

This module manage all the UART ressource.

The e-puck's microcontroller has two integreted UART: UART1 and UART2.

A little exemple to communicate with the e-puck through the uart. For this exemple you have to connect your e-puck to your PC with bluetooth (if you don't know how it works, look at the end of page 3 of this doc: <http://moodle.epfl.ch/mod/resource/view.php?id=12851>). Then open the HyperTerminal with the correct port COM and launch the connection. "Give a character:" should appears on the HyperTerminal.

```
#include <motor_led/e_init_port.h>
#include <uart/e_uart_char.h>

int main(void)
{
    char car;
    e_init_port();
    e_init_uart1();
    e_send_uart1_char("\f\a", 2);           //new page on HyperTerminal
    while(1)
    {
        e_send_uart1_char("Give a character:\r\n", 19);
        // do nothing while the text is not sent and while nothing is coming from the user
        while(e_uart1_sending() || !e_ischar_uart1()) {}
        e_getchar_uart1(&car);           // read the character entered...
        e_send_uart1_char("You have wrote: ", 16);
        e_send_uart1_char(&car, 1);     //... and resend him to uart.
        e_send_uart1_char("\r\n\r\n", 4);
    }
}
```

### Author

Code: Michael Bonani

Doc: Jonathan Besuchet

## 7.88.2 Macro Definition Documentation

7.88.2.1 `#define BAUD115200 7`

7.88.2.2 `#define BAUD230400 3`

7.88.2.3 `#define BAUD460800 1`

7.88.2.4 `#define BAUD921600 0`

## 7.88.3 Function Documentation

7.88.3.1 `int e_getchar_uart1 ( char * car )`

If available, read 1 char and put it in pointer.

### Parameters

<code>car</code>	The pointer where the character will be stored if available
------------------	---

**Returns**

1 if a char has been readed, 0 if no char is available

**7.88.3.2 int e\_getchar\_uart2 ( char \* *car* )**

If available, read 1 char and put it in pointer.

**Parameters**

<i>car</i>	The pointer where the character will be stored if available
------------	---

**Returns**

1 if a char has been readed, 0 if no char is available

**7.88.3.3 void e\_init\_uart1 ( void )**

Init uart 1 at 115200bps, 8 data bits, 1 stop bit, Enable ISR for RX and TX.

**7.88.3.4 void e\_init\_uart2 ( int )**

Init uart 2 at 115200bps, 8 data bits, 1 stop bit, Enable ISR for RX and TX.

**7.88.3.5 int e\_ischar\_uart1 ( )**

Check if something is coming on uart 1.

**Returns**

the number of characters available, 0 if none are available

**7.88.3.6 int e\_ischar\_uart2 ( )**

Check if something is coming on uart 2.

**Returns**

the number of characters available, 0 if none are available

**7.88.3.7 void e\_send\_uart1\_char ( const char \* *buff*, int *length* )**

Send a buffer of char of size length.

## Parameters

<i>buff</i>	The top of the array where the data are stored
<i>length</i>	The length of the array to send

7.88.3.8 void e\_send\_uart2\_char ( const char \* *buff*, int *length* )

Send a buffer of char of size length.

## Parameters

<i>buff</i>	The top of the array where the data are stored
<i>length</i>	The length of the array

## 7.88.3.9 int e\_uart1\_sending ( void )

To check if the sending operation is done.

## Returns

1 if buffer sending is in progress, return 0 if not

## 7.88.3.10 int e\_uart2\_sending ( void )

To check if the sending operation is done.

## Returns

1 if buffer sending is in progress, return 0 if not

## 7.88.4 Variable Documentation

## 7.88.4.1 void\* e\_uart1\_int\_clr\_addr

## 7.88.4.2 int e\_uart1\_int\_clr\_mask

## 7.88.4.3 void\* e\_uart2\_int\_clr\_addr

## 7.88.4.4 int e\_uart2\_int\_clr\_mask

## 7.89 utility/utility.c File Reference

```
#include <motor_led/e_epuck_ports.h>
#include <motor_led/e_init_port.h>
#include <a_d/advance_ad_scan/e_acc.h>
#include <a_d/advance_ad_scan/e_ad_conv.h>
```

## Macros

- #define `MAX_BATT_VALUE` 2560
- #define `MIN_BATT_VALUE` 2070
- #define `BATT_VALUES_RANGE` (`MAX_BATT_VALUE-MIN_BATT_VALUE`)

## Functions

- void `wait` (long `num`)
- int `getselector` ( )
- unsigned int `getBatteryValueRaw` ( )
- unsigned int `getBatteryValuePercentage` ( )
- void `resetTime` (void)
- float `getDiffTimeMs` (void)
- float `getDiffTimeMsAndReset` (void)

## Variables

- int `e_acc_scan` [3][`ACC_SAMP_NB`]
- unsigned int `e_last_acc_scan_id`
- unsigned int `tickAdclsr`

### 7.89.1 Macro Definition Documentation

7.89.1.1 #define `BATT_VALUES_RANGE` (`MAX_BATT_VALUE-MIN_BATT_VALUE`)

7.89.1.2 #define `MAX_BATT_VALUE` 2560

7.89.1.3 #define `MIN_BATT_VALUE` 2070

### 7.89.2 Function Documentation

7.89.2.1 unsigned int `getBatteryValuePercentage` ( )

7.89.2.2 unsigned int `getBatteryValueRaw` ( )

7.89.2.3 float `getDiffTimeMs` ( void )

7.89.2.4 float `getDiffTimeMsAndReset` ( void )

7.89.2.5 int `getselector` ( )

7.89.2.6 void `resetTime` ( void )

7.89.2.7 void `wait` ( long `num` )

### 7.89.3 Variable Documentation

7.89.3.1 int `e_acc_scan`[3][`ACC_SAMP_NB`]

Array to store the acc values

7.89.3.2 unsigned int e\_last\_acc\_scan\_id

7.89.3.3 unsigned int tickAdclsr

## 7.90 utility/utility.h File Reference

### Functions

- void [wait](#) (long num)
- int [getselector](#) ()
- unsigned int [getBatteryValueRaw](#) ()
- unsigned int [getBatteryValuePercentage](#) ()
- void [resetTime](#) (void)
- float [getDiffTimeMs](#) (void)
- float [getDiffTimeMsAndReset](#) (void)

### 7.90.1 Function Documentation

7.90.1.1 unsigned int [getBatteryValuePercentage](#) ( )

7.90.1.2 unsigned int [getBatteryValueRaw](#) ( )

7.90.1.3 float [getDiffTimeMs](#) ( void )

7.90.1.4 float [getDiffTimeMsAndReset](#) ( void )

7.90.1.5 int [getselector](#) ( )

7.90.1.6 void [resetTime](#) ( void )

7.90.1.7 void [wait](#) ( long *num* )





# Index

- `_FBORPOR`
  - `e_init_port.c`, [333](#)
- `_FGS`
  - `e_init_port.c`, [333](#)
- `_FOSC`
  - `e_init_port.c`, [333](#)
- `_FWDT`
  - `e_init_port.c`, [333](#)
- `__attribute__`
  - `advance_ad_scan/e_ad_conv.c`, [50](#)
  - `e_I2C_master_module.c`, [230](#)
  - `e_agenda.c`, [249](#)
  - `e_agenda_fast.c`, [316](#)
  - `e_fft.c`, [226](#)
  - `e_input_signal.c`, [229](#)
  - `e_motors.c`, [294](#)
  - `e_motors_timer3.c`, [336](#)
  - `e_prox.c`, [67](#)
  - `e_prox_timer2.c`, [78](#)
  - `e_remote_control.c`, [311](#)
  - `e_twiddle_factors.c`, [229](#)
  - `fast_2_timer/e_timers.c`, [186](#)
  - `slow_3_timer/e_timers.c`, [189](#)
- `_poxxxx_buffer`
  - `fast_2_timer/e_timers.c`, [187](#)
- `_poxxxx_img_ready`
  - `fast_2_timer/e_timers.c`, [187](#)
- `a_d/advance_ad_scan/e_acc.c`, [39](#)
- `a_d/advance_ad_scan/e_acc.h`, [44](#)
- `a_d/advance_ad_scan/e_ad_conv.c`, [49](#)
- `a_d/advance_ad_scan/e_ad_conv.h`, [53](#)
- `a_d/advance_ad_scan/e_micro.c`, [57](#)
- `a_d/advance_ad_scan/e_micro.h`, [60](#)
- `a_d/advance_ad_scan/e_prox.c`, [63](#)
- `a_d/advance_ad_scan/e_prox.h`, [69](#)
- `a_d/e_accelerometer.c`, [75](#)
- `a_d/e_accelerometer.h`, [76](#)
- `a_d/e_ad_conv.c`, [52](#)
- `a_d/e_ad_conv.h`, [56](#)
- `a_d/e_micro.c`, [59](#)
- `a_d/e_micro.h`, [62](#)
- `a_d/e_prox.c`, [66](#)
- `a_d/e_prox.h`, [72](#)
- `a_d/e_prox_timer2.c`, [77](#)
- `ACC_ADDR`
  - `e_lsm330.c`, [81](#)
- `ACC_PROX_PERIOD`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ACC_PROX_SAMP_FREQ`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ACC_SAMP_NB`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ACCX_BUFFER`
  - `e_acc.h`, [46](#)
- `ACCY_BUFFER`
  - `e_acc.h`, [46](#)
- `ACCZ_BUFFER`
  - `e_acc.h`, [46](#)
- `ACCX`
  - `e_epuck_ports.h`, [327](#)
- `ACCY`
  - `e_epuck_ports.h`, [327](#)
- `ACCZ`
  - `e_epuck_ports.h`, [327](#)
- `ACKNOWLEDGE`
  - `e_I2C_master_module.h`, [234](#)
- `ADC_ISR_PERIOD_MS`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ADCOFF_BASE`
  - `e_po3030k_registers.c`, [142](#)
  - `e_registers.c`, [196](#)
- `ADCS_2_CHAN`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ADCS_3_CHAN`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ADCS_5_CHAN`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ADCS_6_CHAN`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `ADDRHDATA_IN_PACKET_OFFSET`
  - `ComModule.c`, [222](#)
- `ADDRRLDATA_IN_PACKET_OFFSET`
  - `ComModule.c`, [222](#)
- `AESPEED_BASE`
  - `e_po3030k_registers.c`, [142](#)
  - `e_registers.c`, [196](#)
- `AG_ALREADY_CREATED`
  - `e_agenda.h`, [253](#)
  - `e_agenda_fast.h`, [321](#)
- `AG_NOT_FOUND`
  - `e_agenda.h`, [253](#)
  - `e_agenda_fast.h`, [321](#)
- `ALL_ADC`
  - `advance_ad_scan/e_ad_conv.h`, [54](#)
- `AM_MSGTYPE_IN_PACKET_OFFSET`
  - `ComModule.c`, [222](#)
- `AM_MSGTYPE`
  - `ComModule.c`, [222](#)

- ANGLE\_ERROR
  - e\_acc.h, 46
- ARRAY\_HEIGHT
  - e\_poxxxx.h, 182
  - slow\_3\_timer/e\_po3030k.h, 124
- ARRAY\_ORIGINE\_X
  - e\_calc.c, 192
  - e\_calc\_po3030k.c, 99
  - e\_calc\_po6030k.c, 101
  - e\_calc\_po8030d.c, 102
  - e\_po3030k\_registers.c, 142
  - e\_po6030k\_registers.c, 166
  - e\_po8030d\_registers.c, 177
  - e\_registers.c, 196
- ARRAY\_ORIGINE\_Y
  - e\_calc.c, 192
  - e\_calc\_po3030k.c, 99
  - e\_calc\_po6030k.c, 101
  - e\_calc\_po8030d.c, 102
  - e\_po3030k\_registers.c, 142
  - e\_po6030k\_registers.c, 166
  - e\_po8030d\_registers.c, 177
  - e\_registers.c, 196
- ARRAY\_WIDTH
  - e\_poxxxx.h, 182
  - slow\_3\_timer/e\_po3030k.h, 124
- AUDIO\_ON\_DIR
  - e\_epuck\_ports.h, 327
- AUDIO\_ON
  - e\_epuck\_ports.h, 327
- AWBAEENABLE\_BASE
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- AWVAETOL\_BASE
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- acc\_gyro/e\_lsm330.c, 79
- acc\_gyro/e\_lsm330.h, 82
- acc\_x
  - e\_acc.c, 44
  - TypeAccRaw, 37
- acc\_y
  - e\_acc.c, 44
  - TypeAccRaw, 37
- acc\_z
  - e\_acc.c, 44
  - TypeAccRaw, 37
- acceleration
  - e\_acc.c, 44
  - TypeAccSpheric, 38
- activate
  - AgendaType, 34
- address
  - BtDevice, 35
  - BtEPuck, 36
  - e\_remote\_control.c, 312
- address\_temp
  - e\_remote\_control.c, 312
- advance\_ad\_scan/e\_ad\_conv.c
  - \_\_attribute\_\_, 50
  - centre\_z, 51
  - e\_acc\_scan, 51
  - e\_ad\_is\_acquisition\_completed, 50
  - e\_ad\_is\_array\_filled, 50
  - e\_ad\_scan\_off, 51
  - e\_ad\_scan\_on, 51
  - e\_ambient\_and\_reflected\_ir, 51
  - e\_ambient\_ir, 51
  - e\_init\_ad\_scan, 51
  - e\_last\_acc\_scan\_id, 51
  - e\_last\_mic\_scan\_id, 52
  - e\_mic\_scan, 52
  - is\_ad\_acquisition\_completed, 52
  - is\_ad\_array\_filled, 52
  - micro\_only, 52
  - selector, 52
  - tickAdclsr, 52
  - updateAccl2CCounter, 52
- advance\_ad\_scan/e\_ad\_conv.h
  - ACC\_PROX\_PERIOD, 54
  - ACC\_PROX\_SAMP\_FREQ, 54
  - ACC\_SAMP\_NB, 54
  - ADC\_ISR\_PERIOD\_MS, 54
  - ADCS\_2\_CHAN, 54
  - ADCS\_3\_CHAN, 54
  - ADCS\_5\_CHAN, 54
  - ADCS\_6\_CHAN, 54
  - ALL\_ADC, 54
  - e\_ad\_is\_acquisition\_completed, 55
  - e\_ad\_is\_array\_filled, 55
  - e\_ad\_scan\_off, 55
  - e\_ad\_scan\_on, 55
  - e\_init\_ad\_scan, 55
  - MIC\_SAMP\_FREQ, 55
  - MIC\_SAMP\_NB, 55
  - MICRO\_ONLY, 55
  - PULSE LENGHT, 55
  - PULSE\_PERIOD, 55
- advance\_ad\_scan/e\_micro.c
  - e\_get\_micro, 57
  - e\_get\_micro\_average, 58
  - e\_get\_micro\_last\_values, 58
  - e\_get\_micro\_volume, 58
  - e\_last\_mic\_scan\_id, 59
  - e\_mic\_scan, 59
- advance\_ad\_scan/e\_micro.h
  - e\_get\_micro, 61
  - e\_get\_micro\_average, 61
  - e\_get\_micro\_volume, 62
  - MIC0\_BUFFER, 61
  - MIC1\_BUFFER, 61
  - MIC2\_BUFFER, 61
- advance\_ad\_scan/e\_prox.c
  - e\_ambient\_and\_reflected\_ir, 66
  - e\_ambient\_ir, 66
  - e\_calibrate\_ir, 64

- e\_get\_ambient\_light, 64
- e\_get\_calibrated\_prox, 65
- e\_get\_prox, 65
- init\_value\_ir, 66
- advance\_ad\_scan/e\_prox.h
  - e\_calibrate\_ir, 70
  - e\_get\_ambient\_light, 70
  - e\_get\_calibrated\_prox, 70
  - e\_get\_prox, 71
- advance\_one\_timer/e\_led.c
  - e\_blink\_led, 260
  - e\_blink\_led0, 260
  - e\_blink\_led1, 260
  - e\_blink\_led2, 260
  - e\_blink\_led3, 260
  - e\_blink\_led4, 261
  - e\_blink\_led5, 261
  - e\_blink\_led6, 261
  - e\_blink\_led7, 261
  - e\_led\_clear, 261
  - e\_set\_blinking\_cycle, 262
  - e\_set\_body\_led, 262
  - e\_set\_front\_led, 262
  - e\_set\_led, 262
  - e\_start\_led\_blinking, 263
  - e\_stop\_led\_blinking, 263
  - flow\_led, 263
  - k2000\_led, 263
  - LED\_EFFECTS, 259
  - left\_led, 263
  - right\_led, 264
  - snake\_led, 264
- advance\_one\_timer/e\_led.h
  - e\_blink\_led, 273
  - e\_blink\_led0, 273
  - e\_blink\_led1, 273
  - e\_blink\_led2, 273
  - e\_blink\_led3, 273
  - e\_blink\_led4, 274
  - e\_blink\_led5, 274
  - e\_blink\_led6, 274
  - e\_blink\_led7, 274
  - e\_led\_clear, 274
  - e\_set\_body\_led, 275
  - e\_set\_front\_led, 275
  - e\_set\_led, 275
  - e\_start\_led\_blinking, 277
  - e\_stop\_led\_blinking, 277
  - flow\_led, 277
  - k2000\_led, 278
  - left\_led, 278
  - right\_led, 278
  - snake\_led, 278
- advance\_one\_timer/e\_motors.c
  - e\_get\_steps\_left, 286
  - e\_get\_steps\_right, 286
  - e\_init\_motors, 287
  - e\_set\_speed, 287
  - e\_set\_speed\_left, 287
  - e\_set\_speed\_right, 287
  - e\_set\_steps\_left, 288
  - e\_set\_steps\_right, 288
  - left\_motor\_phase, 289
  - left\_speed, 289
  - MAXV, 286
  - nbr\_steps\_left, 289
  - nbr\_steps\_right, 289
  - POWERSAVE, 286
  - right\_motor\_phase, 289
  - right\_speed, 289
  - run\_left\_motor, 288
  - run\_right\_motor, 288
  - TRESHV, 286
- advance\_one\_timer/e\_motors.h
  - e\_get\_steps\_left, 297
  - e\_get\_steps\_right, 297
  - e\_init\_motors, 297
  - e\_set\_speed, 298
  - e\_set\_speed\_left, 298
  - e\_set\_speed\_right, 299
  - e\_set\_steps\_left, 300
  - e\_set\_steps\_right, 301
- advance\_one\_timer/fast\_agenda/e\_led.c
  - e\_blink\_led, 265
  - e\_blink\_led0, 265
  - e\_blink\_led1, 265
  - e\_blink\_led2, 265
  - e\_blink\_led3, 266
  - e\_blink\_led4, 266
  - e\_blink\_led5, 266
  - e\_blink\_led6, 266
  - e\_blink\_led7, 266
  - e\_led\_clear, 267
  - e\_set\_blinking\_cycle, 267
  - e\_set\_body\_led, 267
  - e\_set\_front\_led, 267
  - e\_set\_led, 268
  - e\_start\_led\_blinking, 268
  - e\_stop\_led\_blinking, 268
- advance\_one\_timer/fast\_agenda/e\_led.h
  - e\_blink\_led, 279
  - e\_blink\_led0, 279
  - e\_blink\_led1, 279
  - e\_blink\_led2, 280
  - e\_blink\_led3, 280
  - e\_blink\_led4, 280
  - e\_blink\_led5, 280
  - e\_blink\_led6, 280
  - e\_blink\_led7, 281
  - e\_led\_clear, 281
  - e\_set\_body\_led, 281
  - e\_set\_front\_led, 281
  - e\_set\_led, 282
  - e\_start\_led\_blinking, 282
  - e\_stop\_led\_blinking, 283
- advance\_one\_timer/fast\_agenda/e\_motors.c

- e\_get\_steps\_left, 290
- e\_get\_steps\_right, 290
- e\_init\_motors, 290
- e\_set\_speed, 291
- e\_set\_speed\_left, 291
- e\_set\_speed\_right, 291
- e\_set\_steps\_left, 292
- e\_set\_steps\_right, 292
- left\_motor\_phase, 292
- left\_speed, 292
- MAXV, 290
- nbr\_steps\_left, 292
- nbr\_steps\_right, 292
- POWERSAVE, 290
- right\_motor\_phase, 292
- right\_speed, 293
- run\_left\_motor, 292
- run\_right\_motor, 292
- TRESHV, 290
- advance\_one\_timer/fast\_agenda/e\_motors.h
  - e\_get\_steps\_left, 302
  - e\_get\_steps\_right, 302
  - e\_init\_motors, 302
  - e\_set\_speed, 302
  - e\_set\_speed\_left, 303
  - e\_set\_speed\_right, 304
  - e\_set\_steps\_left, 305
  - e\_set\_steps\_right, 305
- Agenda
  - e\_agenda.h, 253
  - e\_agenda\_fast.h, 321
- agenda\_list
  - e\_agenda.c, 252
  - e\_agenda\_fast.c, 319
- AgendaList, 33
  - agendas, 33
  - motors, 33
  - speed, 33
  - timers\_in\_use, 33
  - waiting, 34
- AgendaType, 34
  - activate, 34
  - counter, 34
  - cycle, 34
  - function, 35
  - next, 35
- agendas
  - AgendaList, 33
- ambient\_and\_reflected\_ir
  - e\_prox.c, 69
  - e\_prox\_timer2.c, 79
- ambient\_ir
  - e\_prox.c, 69
  - e\_prox\_timer2.c, 79
- Analogic/Digital conversion (ADC), 11
- angle\_mem
  - e\_acc.c, 44
- assign\_agenda
  - e\_agenda\_fast.c, 316
- BANK\_REGISTER
  - e\_po6030k\_registers.c, 166
  - e\_po8030d\_registers.c, 177
- BANK\_A
  - e\_po6030k.h, 158
  - e\_po8030d.h, 171
- BANK\_B
  - e\_po6030k.h, 158
  - e\_po8030d.h, 171
- BANK\_C
  - e\_po6030k.h, 158
  - e\_po8030d.h, 171
- BANK\_D
  - e\_po6030k.h, 158
  - e\_po8030d.h, 171
- BASE\_D1
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- BASE\_D2
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- BASE\_D3
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- BASE\_D4
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- BATT\_LOW\_DIR
  - e\_epuck\_ports.h, 327
- BATT\_LOW
  - e\_epuck\_ports.h, 327
- BATT\_VALUES\_RANGE
  - utility.c, 342
- BAUD115200
  - e\_uart\_char.h, 339
- BAUD230400
  - e\_uart\_char.h, 339
- BAUD460800
  - e\_uart\_char.h, 339
- BAUD921600
  - e\_uart\_char.h, 339
- BAYER\_CLOCK\_1
  - e\_po6030k.h, 158
- BAYER\_CLOCK\_2
  - e\_po6030k.h, 158
- BAYER\_CLOCK\_4
  - e\_po6030k.h, 158
- BAYER\_CLOCK\_8
  - e\_po6030k.h, 158
- BIAS\_BASE
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- BODY\_LED\_DIR
  - e\_epuck\_ports.h, 327
- BODY\_LED
  - e\_epuck\_ports.h, 327
- BOTTOMI

- e\_remote\_control.h, 314
- BOTTOMR
  - e\_remote\_control.h, 314
- BRICTR\_BASE
  - e\_po3030k\_registers.c, 142
  - e\_registers.c, 196
- BUFFER\_DATA\_LENGTH
  - ComModule.c, 222
- bbl\_current
  - slow\_3\_timer/e\_timers.c, 190
- blank\_betw\_lines
  - slow\_3\_timer/e\_timers.c, 190
- blank\_row\_betw
  - fast\_2\_timer/e\_timers.c, 188
- Bluetooth, 13
- bluetooth/e\_bluetooth.c, 84
- bluetooth/e\_bluetooth.h, 91
- bpp\_current
  - slow\_3\_timer/e\_timers.c, 190
- BtDevice, 35
  - address, 35
  - class, 35
  - friendly\_name, 36
- BtEPuck, 36
  - address, 36
  - number, 36
- buf\_pos
  - slow\_3\_timer/e\_timers.c, 190
- buffer
  - slow\_3\_timer/e\_timers.c, 190
- bytes\_per\_pixel
  - slow\_3\_timer/e\_timers.c, 190
- CAM\_DATA
  - e\_epuck\_ports.h, 327
- CAM\_HREF\_DIR
  - e\_epuck\_ports.h, 327
- CAM\_HREF
  - e\_epuck\_ports.h, 327
- CAM\_PCLK\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_PCLK
  - e\_epuck\_ports.h, 328
- CAM\_PWDN\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_PWDN
  - e\_epuck\_ports.h, 328
- CAM\_RESET\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_RESET
  - e\_epuck\_ports.h, 328
- CAM\_VSYNC\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_VSYNC
  - e\_epuck\_ports.h, 328
- CAM\_y0
  - e\_epuck\_ports.h, 328
- CAM\_y0\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y1
  - e\_epuck\_ports.h, 328
- CAM\_y1\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y2
  - e\_epuck\_ports.h, 328
- CAM\_y2\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y3
  - e\_epuck\_ports.h, 328
- CAM\_y3\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y4
  - e\_epuck\_ports.h, 328
- CAM\_y4\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y5
  - e\_epuck\_ports.h, 328
- CAM\_y5\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y6
  - e\_epuck\_ports.h, 328
- CAM\_y6\_DIR
  - e\_epuck\_ports.h, 328
- CAM\_y7
  - e\_epuck\_ports.h, 328
- CAM\_y7\_DIR
  - e\_epuck\_ports.h, 329
- CBCRGAIN\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- CHAN\_DOWN
  - e\_remote\_control.h, 314
- CHAN\_UP
  - e\_remote\_control.h, 314
- CLRWDT
  - e\_epuck\_ports.h, 329
- COLGAIN\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- COLOR\_COEF\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- COLOR\_M\_ADDR
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- COM\_MODULE\_DEFAULT\_GROUP
  - ComModule.h, 224
- COM\_MODULE\_HW\_ATTENUATOR\_0DB
  - ComModule.h, 224
- COM\_MODULE\_HW\_ATTENUATOR\_25DB
  - ComModule.h, 224
- COM\_MODULE\_I2C\_ADDR
  - ComModule.c, 222
- COM\_MODULE\_MAXSIZE
  - ComModule.h, 224
- CONFIG\_REG\_ADDR
  - ComModule.c, 222

- CST\_RADIAN
  - e\_acc.h, 46
- calculate\_acc\_raw
  - e\_acc.c, 41
- calculate\_acc\_spherical
  - e\_acc.c, 41
- calibrateGyroscope
  - e\_lsm330.c, 81
  - e\_lsm330.h, 83
- cam\_reg
  - e\_po3030k\_registers.c, 157
  - e\_registers.c, 211
- Camera fast two timers, 14
- Camera slow three timers, 17
- camera/fast\_2\_timer/e\_calc\_po3030k.c, 98
- camera/fast\_2\_timer/e\_calc\_po6030k.c, 100
- camera/fast\_2\_timer/e\_calc\_po8030d.c, 102
- camera/fast\_2\_timer/e\_common.c, 103
- camera/fast\_2\_timer/e\_po3030k.h, 105
- camera/fast\_2\_timer/e\_po3030k\_registers.c, 140
- camera/fast\_2\_timer/e\_po6030k.h, 157
- camera/fast\_2\_timer/e\_po6030k\_registers.c, 165
- camera/fast\_2\_timer/e\_po8030d.h, 170
- camera/fast\_2\_timer/e\_po8030d\_registers.c, 176
- camera/fast\_2\_timer/e\_poxxxx.h, 181
- camera/fast\_2\_timer/e\_timers.c, 185
- camera/slow\_3\_timer/e\_calc.c, 192
- camera/slow\_3\_timer/e\_po3030k.h, 122
- camera/slow\_3\_timer/e\_registers.c, 193
- camera/slow\_3\_timer/e\_timers.c, 188
- camera\_version
  - e\_common.c, 105
- centre\_x
  - e\_acc.c, 44
- centre\_y
  - e\_acc.c, 44
- centre\_z
  - advance\_ad\_scan/e\_ad\_conv.c, 51
  - e\_acc.c, 44
- check
  - e\_remote\_control.c, 312
- check\_temp
  - e\_remote\_control.c, 312
- class
  - BtDevice, 35
- CloseEpuck.m
  - EpuckPort, 241
  - fclose, 241
- codec/e\_common.inc, 211
- codec/e\_sound.c, 211
- codec/e\_sound.h, 212
- ComModule.c
  - ADDRHDATA\_IN\_PACKET\_OFFSET, 222
  - ADDRRLDATA\_IN\_PACKET\_OFFSET, 222
  - AM\_MSGTYPE\_IN\_PACKET\_OFFSET, 222
  - AM\_MSGTYPE, 222
  - BUFFER\_DATA\_LENGTH, 222
  - COM\_MODULE\_I2C\_ADDR, 222
  - CONFIG\_REG\_ADDR, 222
  - FIRSTDATA\_IN\_PACKET\_OFFSET, 222
  - GROUPDATA\_IN\_PACKET\_OFFSET, 222
  - GetHardwareAttenuator, 223
  - GetOwnAddress, 223
  - GetOwnGroup, 223
  - GetRadioEnabledState, 223
  - GetSoftwareAttenuator, 223
  - GetStatus, 223
  - HARDWAREATT\_SET\_FLAG, 222
  - InitComModule, 223
  - IsModulePlugged, 223
  - IsPacketReady, 223
  - OWNADDRH\_REG\_ADDR, 222
  - OWNADDRL\_REG\_ADDR, 222
  - OWNGROUP\_REG\_ADDR, 222
  - PACKET\_LOST\_FLAG, 222
  - PACKET\_READY\_FLAG, 222
  - RADIO\_ENABLED\_FLAG, 222
  - REC\_BUFFER\_END, 222
  - REC\_BUFFER\_START, 222
  - REQUEST\_TO\_SEND\_FLAG, 222
  - ReadRegister, 223
  - SEND\_BUFFER\_END, 222
  - SEND\_BUFFER\_START, 222
  - SEND\_REG\_ADDR, 222
  - SOFTATT\_REG\_ADDR, 222
  - STATUS\_REG\_ADDR, 223
  - SendPacket, 223
  - SetHardwareAttenuator, 223
  - SetOwnAddress, 223
  - SetOwnGroup, 223
  - SetRadioEnabledState, 223
  - SetSoftwareAttenuator, 223
  - TX\_IDLE\_FLAG, 223
  - TX\_SEND\_ERROR, 223
  - TYPEDATA\_IN\_PACKET\_OFFSET, 223
  - WriteRegister, 223
- ComModule.h
  - COM\_MODULE\_DEFAULT\_GROUP, 224
  - COM\_MODULE\_HW\_ATTENUATOR\_0DB, 224
  - COM\_MODULE\_HW\_ATTENUATOR\_25DB, 224
  - COM\_MODULE\_MAXSIZE, 224
  - GetHardwareAttenuator, 224
  - GetOwnGroup, 224
  - GetRadioEnabledState, 225
  - GetSoftwareAttenuator, 225
  - GetStatus, 225
  - InitComModule, 225
  - IsModulePlugged, 225
  - IsPacketReady, 225
  - SendPacket, 225
  - SetHardwareAttenuator, 225
  - SetOwnAddress, 225
  - SetOwnGroup, 225
  - SetRadioEnabledState, 225
  - SetSoftwareAttenuator, 225
- compute\_gcd

- e\_agenda\_fast.c, 316
- continue
  - EpuckGetData.m, 243
- contrib/LIS\_sensors\_turret/e\_devantech.c, 214
- contrib/LIS\_sensors\_turret/e\_devantech.h, 215
- contrib/LIS\_sensors\_turret/e\_sensex.c, 216
- contrib/LIS\_sensors\_turret/e\_sensex.h, 217
- contrib/LIS\_sensors\_turret/e\_sharp.c, 218
- contrib/LIS\_sensors\_turret/e\_sharp.h, 219
- contrib/SWIS\_com\_module/ComModule.c, 220
- contrib/SWIS\_com\_module/ComModule.h, 223
- counter
  - AgendaType, 34
- current\_col
  - slow\_3\_timer/e\_timers.c, 190
- current\_row
  - slow\_3\_timer/e\_timers.c, 191
- cycle
  - AgendaType, 34
- DEVICE\_ID
  - e\_common.c, 104
  - e\_po3030k\_registers.c, 143
  - e\_po6030k\_registers.c, 166
  - e\_po8030d\_registers.c, 177
  - e\_registers.c, 196
- data
  - e\_remote\_control.c, 312
  - EpuckGetData.m, 243
- data\_temp
  - e\_remote\_control.c, 312
- e\_I2C\_master\_module.c
  - \_\_attribute\_\_, 230
  - e\_i2c\_ack, 230
  - e\_i2c\_deinit, 230
  - e\_i2c\_disable, 231
  - e\_i2c\_enable, 231
  - e\_i2c\_init, 231
  - e\_i2c\_mode, 233
  - e\_i2c\_nack, 231
  - e\_i2c\_read, 231
  - e\_i2c\_reset, 232
  - e\_i2c\_restart, 232
  - e\_i2c\_start, 232
  - e\_i2c\_stop, 232
  - e\_i2c\_write, 232
  - e\_interrupts, 233
  - idle\_i2c, 233
- e\_I2C\_master\_module.h
  - ACKNOWLEDGE, 234
  - e\_i2c\_ack, 234
  - e\_i2c\_deinit, 234
  - e\_i2c\_disable, 235
  - e\_i2c\_enable, 235
  - e\_i2c\_init, 235
  - e\_i2c\_nack, 235
  - e\_i2c\_read, 235
  - e\_i2c\_reset, 236
  - e\_i2c\_restart, 236
  - e\_i2c\_start, 236
  - e\_i2c\_stop, 236
  - e\_i2c\_write, 236
  - ERROR, 234
  - OPERATION\_OK, 234
  - READ, 234
  - RESTART, 234
  - START, 234
  - STOP, 234
  - WRITE, 234
- e\_I2C\_protocol.c
  - e\_i2cp\_deinit, 237
  - e\_i2cp\_disable, 237
  - e\_i2cp\_enable, 237
  - e\_i2cp\_init, 238
  - e\_i2cp\_read, 238
  - e\_i2cp\_write, 238
- e\_I2C\_protocol.h
  - e\_i2cp\_deinit, 239
  - e\_i2cp\_disable, 239
  - e\_i2cp\_enable, 239
  - e\_i2cp\_init, 240
  - e\_i2cp\_read, 240
  - e\_i2cp\_read\_string, 240
  - e\_i2cp\_write, 240
  - e\_i2cp\_write\_string, 240
- E\_PO6030K\_SKETCH\_BW
  - e\_po6030k.h, 158
- E\_PO6030K\_SKETCH\_COLOR
  - e\_po6030k.h, 159
- E\_PO8030D\_SKETCH\_BW
  - e\_po8030d.h, 171
- E\_PO8030D\_SKETCH\_COLOR
  - e\_po8030d.h, 171
- e\_acc.c
  - acc\_x, 44
  - acc\_y, 44
  - acc\_z, 44
  - acceleration, 44
  - angle\_mem, 44
  - calculate\_acc\_raw, 41
  - calculate\_acc\_spherical, 41
  - centre\_x, 44
  - centre\_y, 44
  - centre\_z, 44
  - e\_acc\_calibr, 41
  - e\_acc\_scan, 44
  - e\_display\_angle, 41
  - e\_get\_acc, 41
  - e\_get\_acc\_filtered, 42
  - e\_last\_acc\_scan\_id, 44
  - e\_read\_acc, 42
  - e\_read\_acc\_spheric, 42
  - e\_read\_acc\_x, 42
  - e\_read\_acc\_xyz, 42
  - e\_read\_acc\_y, 43
  - e\_read\_acc\_z, 43

- e\_read\_inclination, [43](#)
- e\_read\_orientation, [43](#)
- inclination, [44](#)
- isCalibratingFlag, [44](#)
- lastAccX, [44](#)
- lastAccY, [44](#)
- lastAccZ, [44](#)
- orientation, [44](#)
- updateAccl2CCounter, [44](#)
- e\_acc.h
  - ACCX\_BUFFER, [46](#)
  - ACCY\_BUFFER, [46](#)
  - ACCZ\_BUFFER, [46](#)
  - ANGLE\_ERROR, [46](#)
  - CST\_RADIAN, [46](#)
  - e\_acc\_calibr, [47](#)
  - e\_display\_angle, [47](#)
  - e\_get\_acc, [47](#)
  - e\_get\_acc\_filtered, [47](#)
  - e\_read\_acc, [47](#)
  - e\_read\_acc\_spheric, [48](#)
  - e\_read\_acc\_x, [48](#)
  - e\_read\_acc\_xyz, [48](#)
  - e\_read\_acc\_y, [48](#)
  - e\_read\_acc\_z, [48](#)
  - e\_read\_inclination, [49](#)
  - e\_read\_orientation, [49](#)
  - FILTER\_SIZE, [46](#)
  - GRAVITY\_LSM330, [47](#)
  - GRAVITY, [47](#)
- e\_acc\_calibr
  - e\_acc.c, [41](#)
  - e\_acc.h, [47](#)
- e\_acc\_scan
  - advance\_ad\_scan/e\_ad\_conv.c, [51](#)
  - e\_acc.c, [44](#)
  - utility.c, [342](#)
- e\_accelerometer.c
  - e\_get\_acc, [75](#)
  - e\_init\_acc, [76](#)
- e\_accelerometer.h
  - e\_get\_acc, [77](#)
  - e\_init\_acc, [77](#)
- e\_activate\_agenda
  - e\_agenda.c, [249](#)
  - e\_agenda.h, [253](#)
  - e\_agenda\_fast.c, [316](#)
  - e\_agenda\_fast.h, [321](#)
- e\_activate\_motors
  - e\_agenda\_fast.c, [317](#)
  - e\_agenda\_fast.h, [321](#)
- e\_ad\_conv.c
  - e\_init\_ad, [52](#)
  - e\_read\_ad, [52](#)
- e\_ad\_conv.h
  - e\_init\_ad, [56](#)
  - e\_read\_ad, [56](#)
- e\_ad\_is\_acquisition\_completed
  - advance\_ad\_scan/e\_ad\_conv.c, [50](#)
  - advance\_ad\_scan/e\_ad\_conv.h, [55](#)
- e\_ad\_is\_array\_filled
  - advance\_ad\_scan/e\_ad\_conv.c, [50](#)
  - advance\_ad\_scan/e\_ad\_conv.h, [55](#)
- e\_ad\_scan\_off
  - advance\_ad\_scan/e\_ad\_conv.c, [51](#)
  - advance\_ad\_scan/e\_ad\_conv.h, [55](#)
- e\_ad\_scan\_on
  - advance\_ad\_scan/e\_ad\_conv.c, [51](#)
  - advance\_ad\_scan/e\_ad\_conv.h, [55](#)
- e\_agenda.c
  - \_\_attribute\_\_, [249](#)
  - agenda\_list, [252](#)
  - e\_activate\_agenda, [249](#)
  - e\_destroy\_agenda, [250](#)
  - e\_end\_agendas\_processing, [250](#)
  - e\_pause\_agenda, [250](#)
  - e\_reset\_agenda, [251](#)
  - e\_restart\_agenda, [251](#)
  - e\_set\_agenda\_cycle, [251](#)
  - e\_start\_agendas\_processing, [252](#)
  - EXIT\_OK, [249](#)
- e\_agenda.h
  - AG\_ALREADY\_CREATED, [253](#)
  - AG\_NOT\_FOUND, [253](#)
  - Agenda, [253](#)
  - e\_activate\_agenda, [253](#)
  - e\_destroy\_agenda, [254](#)
  - e\_end\_agendas\_processing, [254](#)
  - e\_pause\_agenda, [254](#)
  - e\_reset\_agenda, [256](#)
  - e\_restart\_agenda, [256](#)
  - e\_set\_agenda\_cycle, [257](#)
  - e\_start\_agendas\_processing, [257](#)
- e\_agenda\_fast.c
  - \_\_attribute\_\_, [316](#)
  - agenda\_list, [319](#)
  - assign\_agenda, [316](#)
  - compute\_gcd, [316](#)
  - e\_activate\_agenda, [316](#)
  - e\_activate\_motors, [317](#)
  - e\_configure\_timer, [317](#)
  - e\_destroy\_agenda, [317](#)
  - e\_end\_agendas\_processing, [317](#)
  - e\_pause\_agenda, [317](#)
  - e\_reset\_agenda, [318](#)
  - e\_restart\_agenda, [318](#)
  - e\_set\_agenda\_cycle, [318](#)
  - e\_set\_motor\_speed, [318](#)
  - e\_set\_timer\_speed, [318](#)
  - e\_start\_agendas\_processing, [318](#)
  - e\_start\_timer\_processing, [318](#)
  - EXIT\_OK, [316](#)
  - find\_function, [319](#)
  - migrate, [319](#)
  - my\_ceil, [319](#)
  - recompute\_speeds, [319](#)



- search\_best\_fit, 319
- e\_agenda\_fast.h
  - AG\_ALREADY\_CREATED, 321
  - AG\_NOT\_FOUND, 321
  - Agenda, 321
  - e\_activate\_agenda, 321
  - e\_activate\_motors, 321
  - e\_configure\_timer, 321
  - e\_destroy\_agenda, 321
  - e\_end\_agendas\_processing, 322
  - e\_pause\_agenda, 322
  - e\_reset\_agenda, 323
  - e\_restart\_agenda, 323
  - e\_set\_agenda\_cycle, 324
  - e\_set\_motor\_speed, 324
  - e\_start\_agendas\_processing, 324
  - e\_start\_timer\_processing, 324
- e\_ambient\_and\_reflected\_ir
  - advance\_ad\_scan/e\_ad\_conv.c, 51
  - advance\_ad\_scan/e\_prox.c, 66
- e\_ambient\_ir
  - advance\_ad\_scan/e\_ad\_conv.c, 51
  - advance\_ad\_scan/e\_prox.c, 66
- e\_blink\_led
  - advance\_one\_timer/e\_led.c, 260
  - advance\_one\_timer/e\_led.h, 273
  - advance\_one\_timer/fast\_agenda/e\_led.c, 265
  - advance\_one\_timer/fast\_agenda/e\_led.h, 279
- e\_blink\_led0
  - advance\_one\_timer/e\_led.c, 260
  - advance\_one\_timer/e\_led.h, 273
  - advance\_one\_timer/fast\_agenda/e\_led.c, 265
  - advance\_one\_timer/fast\_agenda/e\_led.h, 279
- e\_blink\_led1
  - advance\_one\_timer/e\_led.c, 260
  - advance\_one\_timer/e\_led.h, 273
  - advance\_one\_timer/fast\_agenda/e\_led.c, 265
  - advance\_one\_timer/fast\_agenda/e\_led.h, 279
- e\_blink\_led2
  - advance\_one\_timer/e\_led.c, 260
  - advance\_one\_timer/e\_led.h, 273
  - advance\_one\_timer/fast\_agenda/e\_led.c, 265
  - advance\_one\_timer/fast\_agenda/e\_led.h, 280
- e\_blink\_led3
  - advance\_one\_timer/e\_led.c, 260
  - advance\_one\_timer/e\_led.h, 273
  - advance\_one\_timer/fast\_agenda/e\_led.c, 266
  - advance\_one\_timer/fast\_agenda/e\_led.h, 280
- e\_blink\_led4
  - advance\_one\_timer/e\_led.c, 261
  - advance\_one\_timer/e\_led.h, 274
  - advance\_one\_timer/fast\_agenda/e\_led.c, 266
  - advance\_one\_timer/fast\_agenda/e\_led.h, 280
- e\_blink\_led5
  - advance\_one\_timer/e\_led.c, 261
  - advance\_one\_timer/e\_led.h, 274
  - advance\_one\_timer/fast\_agenda/e\_led.c, 266
  - advance\_one\_timer/fast\_agenda/e\_led.h, 280
- e\_blink\_led6
  - advance\_one\_timer/e\_led.c, 261
  - advance\_one\_timer/e\_led.h, 274
  - advance\_one\_timer/fast\_agenda/e\_led.c, 266
  - advance\_one\_timer/fast\_agenda/e\_led.h, 280
- e\_blink\_led7
  - advance\_one\_timer/e\_led.c, 261
  - advance\_one\_timer/e\_led.h, 274
  - advance\_one\_timer/fast\_agenda/e\_led.c, 266
  - advance\_one\_timer/fast\_agenda/e\_led.h, 281
- e\_bluetooth.c
  - e\_bt\_connect\_epuck, 85
  - e\_bt\_establish\_SPP\_link, 85
  - e\_bt\_exit\_tranparent\_mode, 86
  - e\_bt\_factory\_reset, 86
  - e\_bt\_find\_epuck, 86
  - e\_bt\_get\_event\_filter, 86
  - e\_bt\_get\_friendly\_name, 86
  - e\_bt\_inquiry, 87
  - e\_bt\_list\_local\_paired\_device, 87
  - e\_bt\_local\_paired\_device, 91
  - e\_bt\_present\_device, 91
  - e\_bt\_present\_epuck, 91
  - e\_bt\_read\_local\_name, 87
  - e\_bt\_read\_local\_pin\_number, 88
  - e\_bt\_release\_SPP\_link, 88
  - e\_bt\_remove\_local\_paired\_device, 88
  - e\_bt\_reset, 88
  - e\_bt\_send\_SPP\_data, 89
  - e\_bt\_set\_event\_filter, 89
  - e\_bt\_tranparent\_mode, 89
  - e\_bt\_write\_local\_name, 89
  - e\_bt\_write\_local\_pin\_number, 91
  - local\_bt\_PIN, 91
- e\_bluetooth.h
  - e\_bt\_connect\_epuck, 93
  - e\_bt\_establish\_SPP\_link, 93
  - e\_bt\_exit\_tranparent\_mode, 93
  - e\_bt\_factory\_reset, 93
  - e\_bt\_find\_epuck, 93
  - e\_bt\_get\_event\_filter, 94
  - e\_bt\_get\_friendly\_name, 94
  - e\_bt\_inquiry, 94
  - e\_bt\_list\_local\_paired\_device, 95
  - e\_bt\_local\_paired\_device, 98
  - e\_bt\_present\_device, 98
  - e\_bt\_present\_epuck, 98
  - e\_bt\_read\_local\_name, 95
  - e\_bt\_read\_local\_pin\_number, 95
  - e\_bt\_release\_SPP\_link, 95
  - e\_bt\_remove\_local\_paired\_device, 96
  - e\_bt\_reset, 96
  - e\_bt\_send\_SPP\_data, 96
  - e\_bt\_set\_event\_filter, 96
  - e\_bt\_tranparent\_mode, 97
  - e\_bt\_write\_local\_name, 97
  - e\_bt\_write\_local\_pin\_number, 97
- e\_bt\_connect\_epuck

- e\_bluetooth.c, 85
- e\_bluetooth.h, 93
- e\_bt\_establish\_SPP\_link
  - e\_bluetooth.c, 85
  - e\_bluetooth.h, 93
- e\_bt\_exit\_tranparent\_mode
  - e\_bluetooth.c, 86
  - e\_bluetooth.h, 93
- e\_bt\_factory\_reset
  - e\_bluetooth.c, 86
  - e\_bluetooth.h, 93
- e\_bt\_find\_epuck
  - e\_bluetooth.c, 86
  - e\_bluetooth.h, 93
- e\_bt\_get\_event\_filter
  - e\_bluetooth.c, 86
  - e\_bluetooth.h, 94
- e\_bt\_get\_friendly\_name
  - e\_bluetooth.c, 86
  - e\_bluetooth.h, 94
- e\_bt\_inquiry
  - e\_bluetooth.c, 87
  - e\_bluetooth.h, 94
- e\_bt\_list\_local\_paired\_device
  - e\_bluetooth.c, 87
  - e\_bluetooth.h, 95
- e\_bt\_local\_paired\_device
  - e\_bluetooth.c, 91
  - e\_bluetooth.h, 98
- e\_bt\_present\_device
  - e\_bluetooth.c, 91
  - e\_bluetooth.h, 98
- e\_bt\_present\_epuck
  - e\_bluetooth.c, 91
  - e\_bluetooth.h, 98
- e\_bt\_read\_local\_name
  - e\_bluetooth.c, 87
  - e\_bluetooth.h, 95
- e\_bt\_read\_local\_pin\_number
  - e\_bluetooth.c, 88
  - e\_bluetooth.h, 95
- e\_bt\_release\_SPP\_link
  - e\_bluetooth.c, 88
  - e\_bluetooth.h, 95
- e\_bt\_remove\_local\_paired\_device
  - e\_bluetooth.c, 88
  - e\_bluetooth.h, 96
- e\_bt\_reset
  - e\_bluetooth.c, 88
  - e\_bluetooth.h, 96
- e\_bt\_send\_SPP\_data
  - e\_bluetooth.c, 89
  - e\_bluetooth.h, 96
- e\_bt\_set\_event\_filter
  - e\_bluetooth.c, 89
  - e\_bluetooth.h, 96
- e\_bt\_tranparent\_mode
  - e\_bluetooth.c, 89
- e\_bluetooth.h, 97
- e\_bt\_write\_local\_name
  - e\_bluetooth.c, 89
  - e\_bluetooth.h, 97
- e\_bt\_write\_local\_pin\_number
  - e\_bluetooth.c, 91
  - e\_bluetooth.h, 97
- e\_calc.c
  - ARRAY\_ORIGINE\_X, 192
  - ARRAY\_ORIGINE\_Y, 192
  - e\_po3030k\_config\_cam, 192
  - e\_po3030k\_get\_bytes\_per\_pixel, 193
  - e\_po3030k\_init\_cam, 193
- e\_calc\_po3030k.c
  - ARRAY\_ORIGINE\_X, 99
  - ARRAY\_ORIGINE\_Y, 99
  - e\_po3030k\_config\_cam, 99
  - e\_po3030k\_get\_bytes\_per\_pixel, 99
  - IRQ\_PIX\_LAT, 99
- e\_calc\_po6030k.c
  - ARRAY\_ORIGINE\_X, 101
  - ARRAY\_ORIGINE\_Y, 101
  - e\_po6030k\_config\_cam, 101
  - e\_po6030k\_get\_bytes\_per\_pixel, 101
  - IRQ\_PIX\_LAT, 101
- e\_calc\_po8030d.c
  - ARRAY\_ORIGINE\_X, 102
  - ARRAY\_ORIGINE\_Y, 102
  - e\_po8030d\_config\_cam, 102
  - e\_po8030d\_get\_bytes\_per\_pixel, 103
  - IRQ\_PIX\_LAT, 102
- e\_calibrate\_ir
  - advance\_ad\_scan/e\_prox.c, 64
  - advance\_ad\_scan/e\_prox.h, 70
- e\_close\_sound
  - e\_sound.c, 211
  - e\_sound.h, 212
- e\_common.c
  - camera\_version, 105
  - DEVICE\_ID, 104
  - e\_poxxxx\_config\_cam, 104
  - e\_poxxxx\_get\_orientation, 104
  - e\_poxxxx\_init\_cam, 104
  - e\_poxxxx\_set\_awb\_ae, 104
  - e\_poxxxx\_set\_exposure, 104
  - e\_poxxxx\_set\_mirror, 105
  - e\_poxxxx\_set\_rgb\_gain, 105
  - e\_poxxxx\_write\_cam\_registers, 105
  - getCameraVersion, 105
  - readee, 105
- e\_configure\_timer
  - e\_agenda\_fast.c, 317
  - e\_agenda\_fast.h, 321
- e\_destroy\_agenda
  - e\_agenda.c, 250
  - e\_agenda.h, 254
  - e\_agenda\_fast.c, 317
  - e\_agenda\_fast.h, 321

- e\_devantech.c
  - e\_disable\_devantech, 214
  - e\_get\_delay\_devantech, 214
  - e\_get\_dist\_devantech, 214
  - e\_get\_light\_devantech, 214
  - e\_get\_sr\_devantech, 214
  - e\_i2cd\_readb, 214
  - e\_i2cd\_write, 214
  - e\_init\_devantech, 215
  - e\_set\_gain\_devantech, 215
  - e\_set\_range\_devantech, 215
- e\_devantech.h
  - e\_disable\_devantech, 216
  - e\_get\_delay\_devantech, 216
  - e\_get\_dist\_devantech, 216
  - e\_get\_light\_devantech, 216
  - e\_get\_sr\_devantech, 216
  - e\_i2cd\_readb, 216
  - e\_i2cd\_readw, 216
  - e\_i2cd\_write, 216
  - e\_init\_devantech, 216
  - e\_set\_gain\_devantech, 216
  - e\_set\_range\_devantech, 216
- e\_disable\_devantech
  - e\_devantech.c, 214
  - e\_devantech.h, 216
- e\_display\_angle
  - e\_acc.c, 41
  - e\_acc.h, 47
- e\_doFFT\_asm
  - e\_fft.c, 226
  - e\_fft.h, 227
- e\_end\_agendas\_processing
  - e\_agenda.c, 250
  - e\_agenda.h, 254
  - e\_agenda\_fast.c, 317
  - e\_agenda\_fast.h, 322
- e\_epuck\_ports.h
  - ACCX, 327
  - ACCY, 327
  - ACCZ, 327
  - AUDIO\_ON\_DIR, 327
  - AUDIO\_ON, 327
  - BATT\_LOW\_DIR, 327
  - BATT\_LOW, 327
  - BODY\_LED\_DIR, 327
  - BODY\_LED, 327
  - CAM\_DATA, 327
  - CAM\_HREF\_DIR, 327
  - CAM\_HREF, 327
  - CAM\_PCLK\_DIR, 328
  - CAM\_PCLK, 328
  - CAM\_PWDN\_DIR, 328
  - CAM\_PWDN, 328
  - CAM\_RESET\_DIR, 328
  - CAM\_RESET, 328
  - CAM\_VSYNC\_DIR, 328
  - CAM\_VSYNC, 328
  - CAM\_y0, 328
  - CAM\_y0\_DIR, 328
  - CAM\_y1, 328
  - CAM\_y1\_DIR, 328
  - CAM\_y2, 328
  - CAM\_y2\_DIR, 328
  - CAM\_y3, 328
  - CAM\_y3\_DIR, 328
  - CAM\_y4, 328
  - CAM\_y4\_DIR, 328
  - CAM\_y5, 328
  - CAM\_y5\_DIR, 328
  - CAM\_y6, 328
  - CAM\_y6\_DIR, 328
  - CAM\_y7, 328
  - CAM\_y7\_DIR, 329
  - CLRWDT, 329
  - FALSE, 329
  - FCY, 329
  - FOSC, 329
  - FRONT\_LED\_DIR, 329
  - FRONT\_LED, 329
  - IDLE, 329
  - INPUT\_PIN, 329
  - INTERRUPT\_DELAY, 329
  - INTERRUPT\_OFF, 329
  - INTERRUPT\_ON, 329
  - IR0, 329
  - IR1, 329
  - IR2, 329
  - IR3, 329
  - IR4, 329
  - IR5, 329
  - IR6, 329
  - IR7, 329
  - LED0, 329
  - LED0\_DIR, 329
  - LED1, 329
  - LED1\_DIR, 330
  - LED2, 330
  - LED2\_DIR, 330
  - LED3, 330
  - LED3\_DIR, 330
  - LED4, 330
  - LED4\_DIR, 330
  - LED5, 330
  - LED5\_DIR, 330
  - LED6, 330
  - LED6\_DIR, 330
  - LED7, 330
  - LED7\_DIR, 330
  - MIC1, 330
  - MIC2, 330
  - MIC3, 330
  - MICROSEC, 330
  - MILLISEC, 330
  - MOTOR1\_PHA\_DIR, 330
  - MOTOR1\_PHB\_DIR, 330

- MOTOR1\_PHC\_DIR, [331](#)
- MOTOR1\_PHD\_DIR, [331](#)
- MOTOR1\_PHA, [330](#)
- MOTOR1\_PHB, [330](#)
- MOTOR1\_PHC, [330](#)
- MOTOR1\_PHD, [331](#)
- MOTOR2\_PHA\_DIR, [331](#)
- MOTOR2\_PHB\_DIR, [331](#)
- MOTOR2\_PHC\_DIR, [331](#)
- MOTOR2\_PHD\_DIR, [331](#)
- MOTOR2\_PHA, [331](#)
- MOTOR2\_PHB, [331](#)
- MOTOR2\_PHC, [331](#)
- MOTOR2\_PHD, [331](#)
- NANOSEC, [331](#)
- NOP, [331](#)
- OUTPUT\_PIN, [331](#)
- PLL, [331](#)
- PULSE\_IR0, [331](#)
- PULSE\_IR0\_DIR, [331](#)
- PULSE\_IR1, [331](#)
- PULSE\_IR1\_DIR, [331](#)
- PULSE\_IR2, [331](#)
- PULSE\_IR2\_DIR, [331](#)
- PULSE\_IR3, [331](#)
- PULSE\_IR3\_DIR, [331](#)
- REMOTE\_DIR, [332](#)
- REMOTE, [332](#)
- RESET, [332](#)
- SELECTOR0, [332](#)
- SELECTOR0\_DIR, [332](#)
- SELECTOR1, [332](#)
- SELECTOR1\_DIR, [332](#)
- SELECTOR2, [332](#)
- SELECTOR2\_DIR, [332](#)
- SELECTOR3, [332](#)
- SELECTOR3\_DIR, [332](#)
- SIO\_C\_DIR, [332](#)
- SIO\_D\_DIR, [332](#)
- SIO\_C, [332](#)
- SIO\_D, [332](#)
- SLEEP, [332](#)
- STOP\_TMR1, [332](#)
- STOP\_TMR2, [332](#)
- STOP\_TMR3, [332](#)
- STOP\_TMR4, [332](#)
- STOP\_TMR5, [332](#)
- TCY\_PIC, [332](#)
- TRUE, [332](#)
- e\_fast\_copy
  - e\_fft\_utilities.h, [227](#)
- e\_fft.c
  - \_\_attribute\_\_, [226](#)
  - e\_doFFT\_asm, [226](#)
- e\_fft.h
  - e\_doFFT\_asm, [227](#)
  - FFT\_BLOCK\_LENGTH, [227](#)
  - LOG2\_BLOCK\_LENGTH, [227](#)
- e\_fft\_utilities.h
  - e\_fast\_copy, [227](#)
  - e\_subtract\_mean, [228](#)
- e\_get\_acc
  - e\_acc.c, [41](#)
  - e\_acc.h, [47](#)
  - e\_accelerometer.c, [75](#)
  - e\_accelerometer.h, [77](#)
- e\_get\_acc\_filtered
  - e\_acc.c, [42](#)
  - e\_acc.h, [47](#)
- e\_get\_address
  - e\_remote\_control.c, [311](#)
  - e\_remote\_control.h, [314](#)
- e\_get\_ambient\_light
  - advance\_ad\_scan/e\_prox.c, [64](#)
  - advance\_ad\_scan/e\_prox.h, [70](#)
  - e\_prox.c, [67](#)
  - e\_prox.h, [73](#)
  - e\_prox\_timer2.c, [78](#)
- e\_get\_calibrated\_prox
  - advance\_ad\_scan/e\_prox.c, [65](#)
  - advance\_ad\_scan/e\_prox.h, [70](#)
- e\_get\_check
  - e\_remote\_control.c, [311](#)
  - e\_remote\_control.h, [314](#)
- e\_get\_data
  - e\_remote\_control.c, [311](#)
  - e\_remote\_control.h, [314](#)
- e\_get\_delay\_devantech
  - e\_devantech.c, [214](#)
  - e\_devantech.h, [216](#)
- e\_get\_dist\_devantech
  - e\_devantech.c, [214](#)
  - e\_devantech.h, [216](#)
- e\_get\_dist\_sharp
  - e\_sharp.c, [219](#)
  - e\_sharp.h, [220](#)
- e\_get\_light\_devantech
  - e\_devantech.c, [214](#)
  - e\_devantech.h, [216](#)
- e\_get\_micro
  - advance\_ad\_scan/e\_micro.c, [57](#)
  - advance\_ad\_scan/e\_micro.h, [61](#)
  - e\_micro.c, [60](#)
  - e\_micro.h, [63](#)
- e\_get\_micro\_average
  - advance\_ad\_scan/e\_micro.c, [58](#)
  - advance\_ad\_scan/e\_micro.h, [61](#)
- e\_get\_micro\_last\_values
  - advance\_ad\_scan/e\_micro.c, [58](#)
- e\_get\_micro\_volume
  - advance\_ad\_scan/e\_micro.c, [58](#)
  - advance\_ad\_scan/e\_micro.h, [62](#)
- e\_get\_prox
  - advance\_ad\_scan/e\_prox.c, [65](#)
  - advance\_ad\_scan/e\_prox.h, [71](#)
  - e\_prox.c, [68](#)

- e\_prox.h, 73
- e\_prox\_timer2.c, 78
- e\_get\_sensex\_t\_wait
  - e\_sensex\_t.c, 217
  - e\_sensex\_t.h, 218
- e\_get\_sr\_devantech
  - e\_devantech.c, 214
  - e\_devantech.h, 216
- e\_get\_steps\_left
  - advance\_one\_timer/e\_motors.c, 286
  - advance\_one\_timer/e\_motors.h, 297
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 290
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 302
  - e\_motors.c, 294
  - e\_motors.h, 306
  - e\_motors\_timer3.c, 336
- e\_get\_steps\_right
  - advance\_one\_timer/e\_motors.c, 286
  - advance\_one\_timer/e\_motors.h, 297
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 290
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 302
  - e\_motors.c, 294
  - e\_motors.h, 306
  - e\_motors\_timer3.c, 336
- e\_getchar\_uart1
  - e\_uart\_char.h, 339
- e\_getchar\_uart2
  - e\_uart\_char.h, 340
- e\_i2c\_ack
  - e\_I2C\_master\_module.c, 230
  - e\_I2C\_master\_module.h, 234
- e\_i2c\_deinit
  - e\_I2C\_master\_module.c, 230
  - e\_I2C\_master\_module.h, 234
- e\_i2c\_disable
  - e\_I2C\_master\_module.c, 231
  - e\_I2C\_master\_module.h, 235
- e\_i2c\_enable
  - e\_I2C\_master\_module.c, 231
  - e\_I2C\_master\_module.h, 235
- e\_i2c\_init
  - e\_I2C\_master\_module.c, 231
  - e\_I2C\_master\_module.h, 235
- e\_i2c\_mode
  - e\_I2C\_master\_module.c, 233
- e\_i2c\_nack
  - e\_I2C\_master\_module.c, 231
  - e\_I2C\_master\_module.h, 235
- e\_i2c\_read
  - e\_I2C\_master\_module.c, 231
  - e\_I2C\_master\_module.h, 235
- e\_i2c\_reset
  - e\_I2C\_master\_module.c, 232
  - e\_I2C\_master\_module.h, 236
- e\_i2c\_restart
  - e\_I2C\_master\_module.c, 232
  - e\_I2C\_master\_module.h, 236
- e\_i2c\_start
  - e\_I2C\_master\_module.c, 232
  - e\_I2C\_master\_module.h, 236
- e\_i2c\_stop
  - e\_I2C\_master\_module.c, 232
  - e\_I2C\_master\_module.h, 236
- e\_i2c\_write
  - e\_I2C\_master\_module.c, 232
  - e\_I2C\_master\_module.h, 236
- e\_i2cd\_readb
  - e\_devantech.c, 214
  - e\_devantech.h, 216
- e\_i2cd\_readw
  - e\_devantech.h, 216
- e\_i2cd\_write
  - e\_devantech.c, 214
  - e\_devantech.h, 216
- e\_i2cp\_deinit
  - e\_I2C\_protocol.c, 237
  - e\_I2C\_protocol.h, 239
- e\_i2cp\_disable
  - e\_I2C\_protocol.c, 237
  - e\_I2C\_protocol.h, 239
- e\_i2cp\_enable
  - e\_I2C\_protocol.c, 237
  - e\_I2C\_protocol.h, 239
- e\_i2cp\_init
  - e\_I2C\_protocol.c, 238
  - e\_I2C\_protocol.h, 240
- e\_i2cp\_read
  - e\_I2C\_protocol.c, 238
  - e\_I2C\_protocol.h, 240
- e\_i2cp\_read\_string
  - e\_I2C\_protocol.h, 240
- e\_i2cp\_write
  - e\_I2C\_protocol.c, 238
  - e\_I2C\_protocol.h, 240
- e\_i2cp\_write\_string
  - e\_I2C\_protocol.h, 240
- e\_init\_acc
  - e\_accelerometer.c, 76
  - e\_accelerometer.h, 77
- e\_init\_ad
  - e\_ad\_conv.c, 52
  - e\_ad\_conv.h, 56
- e\_init\_ad\_scan
  - advance\_ad\_scan/e\_ad\_conv.c, 51
  - advance\_ad\_scan/e\_ad\_conv.h, 55
- e\_init\_codec\_slave
  - e\_sound.h, 212
- e\_init\_dci\_master
  - e\_sound.h, 213
- e\_init\_devantech
  - e\_devantech.c, 215
  - e\_devantech.h, 216
- e\_init\_micro
  - e\_micro.c, 60
  - e\_micro.h, 63
- e\_init\_motors

- advance\_one\_timer/e\_motors.c, 287
- advance\_one\_timer/e\_motors.h, 297
- advance\_one\_timer/fast\_agenda/e\_motors.c, 290
- advance\_one\_timer/fast\_agenda/e\_motors.h, 302
- e\_motors.c, 294
- e\_motors.h, 306
- e\_motors\_timer3.c, 336
- e\_init\_port
  - e\_init\_port.c, 333
  - e\_init\_port.h, 334
- e\_init\_port.c
  - \_FBORPOR, 333
  - \_FGS, 333
  - \_FOSC, 333
  - \_FWDT, 333
  - e\_init\_port, 333
  - isEpuckVersion1\_3, 333
  - isPresentFlag, 334
  - testAccGyroPresence, 334
- e\_init\_port.h
  - e\_init\_port, 334
  - isEpuckVersion1\_3, 334
- e\_init\_prox
  - e\_prox.c, 68
  - e\_prox.h, 74
  - e\_prox\_timer2.c, 79
- e\_init\_remote\_control
  - e\_remote\_control.c, 311
  - e\_remote\_control.h, 314
- e\_init\_sensex
  - e\_sensex.c, 217
  - e\_sensex.h, 218
- e\_init\_sharp
  - e\_sharp.c, 219
  - e\_sharp.h, 220
- e\_init\_sound
  - e\_sound.c, 211
  - e\_sound.h, 213
- e\_init\_uart1
  - e\_uart\_char.h, 340
- e\_init\_uart2
  - e\_uart\_char.h, 340
- e\_input\_signal.c
  - \_\_attribute\_\_, 229
- e\_interrupts
  - e\_I2C\_master\_module.c, 233
- e\_ischar\_uart1
  - e\_uart\_char.h, 340
- e\_ischar\_uart2
  - e\_uart\_char.h, 340
- e\_last\_acc\_scan\_id
  - advance\_ad\_scan/e\_ad\_conv.c, 51
  - e\_acc.c, 44
  - utility.c, 342
- e\_last\_mic\_scan\_id
  - advance\_ad\_scan/e\_ad\_conv.c, 52
  - advance\_ad\_scan/e\_micro.c, 59
- e\_led.c
  - e\_led\_clear, 269
  - e\_set\_body\_led, 269
  - e\_set\_front\_led, 271
  - e\_set\_led, 271
- e\_led.h
  - e\_led\_clear, 284
  - e\_set\_body\_led, 284
  - e\_set\_front\_led, 284
  - e\_set\_led, 284
- e\_led\_clear
  - advance\_one\_timer/e\_led.c, 261
  - advance\_one\_timer/e\_led.h, 274
  - advance\_one\_timer/fast\_agenda/e\_led.c, 267
  - advance\_one\_timer/fast\_agenda/e\_led.h, 281
  - e\_led.c, 269
  - e\_led.h, 284
- e\_lsm330.c
  - ACC\_ADDR, 81
  - calibrateGyroscope, 81
  - GYRO\_ADDR, 81
  - getAllAxesAcc, 81
  - getAllAxesAccRaw, 81
  - getAllAxesGyro, 81
  - getAllAxesGyroRaw, 81
  - getTemperature, 81
  - getXAxisAcc, 81
  - getXAxisGyro, 81
  - getYAxisAcc, 81
  - getYAxisGyro, 81
  - getZAxisAcc, 81
  - getZAxisGyro, 81
  - gyroOffsetX, 82
  - gyroOffsetY, 82
  - gyroOffsetZ, 82
  - initAccAndGyro, 81
  - rawToDpms, 81
  - rawToDps, 81
  - readReg, 81
  - readRegMulti, 82
  - writeReg, 82
- e\_lsm330.h
  - calibrateGyroscope, 83
  - getAllAxesAcc, 83
  - getAllAxesAccRaw, 83
  - getAllAxesGyro, 83
  - getAllAxesGyroRaw, 83
  - getTemperature, 83
  - getXAxisAcc, 83
  - getXAxisGyro, 83
  - getYAxisAcc, 83
  - getYAxisGyro, 84
  - getZAxisAcc, 84
  - getZAxisGyro, 84
  - initAccAndGyro, 84
  - rawToDpms, 84
  - rawToDps, 84
- e\_mic\_scan
  - advance\_ad\_scan/e\_ad\_conv.c, 52

- advance\_ad\_scan/e\_micro.c, 59
- e\_micro.c
  - e\_get\_micro, 60
  - e\_init\_micro, 60
- e\_micro.h
  - e\_get\_micro, 63
  - e\_init\_micro, 63
- e\_motors.c
  - \_\_attribute\_\_, 294
  - e\_get\_steps\_left, 294
  - e\_get\_steps\_right, 294
  - e\_init\_motors, 294
  - e\_set\_speed\_left, 295
  - e\_set\_speed\_right, 295
  - e\_set\_steps\_left, 295
  - e\_set\_steps\_right, 296
  - left\_speed, 296
  - nbr\_pas\_left, 296
  - nbr\_pas\_right, 296
  - right\_speed, 296
- e\_motors.h
  - e\_get\_steps\_left, 306
  - e\_get\_steps\_right, 306
  - e\_init\_motors, 306
  - e\_set\_speed\_left, 307
  - e\_set\_speed\_right, 308
  - e\_set\_steps\_left, 309
  - e\_set\_steps\_right, 309
- e\_motors\_timer3.c
  - \_\_attribute\_\_, 336
  - e\_get\_steps\_left, 336
  - e\_get\_steps\_right, 336
  - e\_init\_motors, 336
  - e\_set\_speed\_left, 336
  - e\_set\_speed\_right, 336
  - e\_set\_steps\_left, 337
  - e\_set\_steps\_right, 337
  - left\_speed, 337
  - motor\_counter\_left, 337
  - motor\_counter\_left\_init, 337
  - motor\_counter\_right, 337
  - motor\_counter\_right\_init, 337
  - nbr\_pas\_left, 337
  - nbr\_pas\_right, 337
  - right\_speed, 338
- e\_pause\_agenda
  - e\_agenda.c, 250
  - e\_agenda.h, 254
  - e\_agenda\_fast.c, 317
  - e\_agenda\_fast.h, 322
- e\_play\_sound
  - e\_sound.c, 211
  - e\_sound.h, 213
- e\_po3030k\_SetColorMatrix
  - e\_po3030k\_registers.c, 156
  - e\_registers.c, 209
- e\_po3030k\_apply\_timer\_config
  - slow\_3\_timer/e\_po3030k.h, 125
  - slow\_3\_timer/e\_timers.c, 189
- e\_po3030k\_config\_cam
  - e\_calc.c, 192
  - e\_calc\_po3030k.c, 99
  - fast\_2\_timer/e\_po3030k.h, 107
  - slow\_3\_timer/e\_po3030k.h, 125
- e\_po3030k\_get\_bytes\_per\_pixel
  - e\_calc.c, 193
  - e\_calc\_po3030k.c, 99
  - fast\_2\_timer/e\_po3030k.h, 108
  - slow\_3\_timer/e\_po3030k.h, 126
- e\_po3030k\_get\_register
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 198
  - fast\_2\_timer/e\_po3030k.h, 109
  - slow\_3\_timer/e\_po3030k.h, 127
- e\_po3030k\_init\_cam
  - e\_calc.c, 193
  - fast\_2\_timer/e\_po3030k.h, 109
  - slow\_3\_timer/e\_po3030k.h, 127
- e\_po3030k\_is\_img\_ready
  - slow\_3\_timer/e\_po3030k.h, 127
  - slow\_3\_timer/e\_timers.c, 189
- e\_po3030k\_launch\_capture
  - slow\_3\_timer/e\_po3030k.h, 127
  - slow\_3\_timer/e\_timers.c, 189
- e\_po3030k\_read\_cam\_registers
  - e\_po3030k\_registers.c, 145
  - e\_registers.c, 198
  - fast\_2\_timer/e\_po3030k.h, 109
  - slow\_3\_timer/e\_po3030k.h, 128
- e\_po3030k\_registers.c
  - ADCOFF\_BASE, 142
  - AESPEED\_BASE, 142
  - ARRAY\_ORIGINE\_X, 142
  - ARRAY\_ORIGINE\_Y, 142
  - AWBAEENABLE\_BASE, 142
  - AWVAETOL\_BASE, 142
  - BASE\_D1, 142
  - BASE\_D2, 142
  - BASE\_D3, 142
  - BASE\_D4, 142
  - BIAS\_BASE, 142
  - BRICTR\_BASE, 142
  - CBCRGAIN\_BASE, 143
  - COLGAIN\_BASE, 143
  - COLOR\_COEF\_BASE, 143
  - COLOR\_M\_ADDR, 143
  - cam\_reg, 157
  - DEVICE\_ID, 143
  - e\_po3030k\_SetColorMatrix, 156
  - e\_po3030k\_get\_register, 144
  - e\_po3030k\_read\_cam\_registers, 145
  - e\_po3030k\_set\_adc\_offset, 145
  - e\_po3030k\_set\_ae\_speed, 145
  - e\_po3030k\_set\_awb\_ae, 145
  - e\_po3030k\_set\_awb\_ae\_tol, 146
  - e\_po3030k\_set\_bias, 146

e\_po3030k\_set\_brigh\_contr, 146  
 e\_po3030k\_set\_cb\_cr\_gain, 147  
 e\_po3030k\_set\_color\_gain, 147  
 e\_po3030k\_set\_color\_mode, 147  
 e\_po3030k\_set\_edge\_prop, 148  
 e\_po3030k\_set\_exposure, 148  
 e\_po3030k\_set\_flicker\_detection, 149  
 e\_po3030k\_set\_flicker\_man\_set, 149  
 e\_po3030k\_set\_flicker\_mode, 149  
 e\_po3030k\_set\_gamma\_coef, 150  
 e\_po3030k\_set\_integr\_time, 150  
 e\_po3030k\_set\_lens\_gain, 150  
 e\_po3030k\_set\_max\_min\_awb, 151  
 e\_po3030k\_set\_max\_min\_exp, 151  
 e\_po3030k\_set\_mirror, 151  
 e\_po3030k\_set\_ref\_exposure, 152  
 e\_po3030k\_set\_register, 152  
 e\_po3030k\_set\_sampling\_mode, 152  
 e\_po3030k\_set\_sepia, 153  
 e\_po3030k\_set\_sepia\_tone, 153  
 e\_po3030k\_set\_speed, 153  
 e\_po3030k\_set\_vsync, 154  
 e\_po3030k\_set\_weight\_win, 154  
 e\_po3030k\_set\_ww, 155  
 e\_po3030k\_set\_wx, 155  
 e\_po3030k\_set\_wy, 155  
 e\_po3030k\_sync\_register\_array, 156  
 e\_po3030k\_write\_cam\_registers, 156  
 e\_po3030k\_write\_gamma\_coef, 157  
 EDGE\_BASE, 143  
 EXPOSURE\_BASE, 143  
 FLICKM\_BASE, 143  
 FLICKP\_BASE, 143  
 FRAME\_HEIGHT, 143  
 FRAME\_WIDTH, 143  
 GAMMA\_BASE, 143  
 GAMMASELCOL\_BASE, 143  
 INTEGR\_BASE, 143  
 LENS\_G\_BASE, 143  
 MCLK\_P\_NS, 143  
 MCLK, 143  
 MINMAXAWB\_BASE, 143  
 MINMAXEXP\_BASE, 143  
 MIRROR\_BASE, 143  
 MODE\_GRAYSCALE, 143  
 MODE\_R5G6B5, 143  
 MODE\_YUV, 143  
 NB\_REGISTERS, 144  
 po3030k\_get\_pixelclock, 157  
 REFREPO\_BASE, 144  
 SAMPLING\_ADDR, 144  
 SEPIA\_BASE, 144  
 SEPIATONE\_BASE, 144  
 SPEED\_ADDR, 144  
 VSYNCCOL\_BASE, 144  
 VSYNCSTART\_BASE, 144  
 VSYNCSTOP\_BASE, 144  
 WEIGHWIN\_BASE, 144  
 WINDOW\_X1\_BASE, 144  
 WINDOW\_X2\_BASE, 144  
 WINDOW\_Y1\_BASE, 144  
 WINDOW\_Y2\_BASE, 144  
 WW\_BASE, 144  
 e\_po3030k\_set\_adc\_offset  
   e\_po3030k\_registers.c, 145  
   e\_registers.c, 198  
   fast\_2\_timer/e\_po3030k.h, 109  
   slow\_3\_timer/e\_po3030k.h, 128  
 e\_po3030k\_set\_ae\_speed  
   e\_po3030k\_registers.c, 145  
   e\_registers.c, 199  
   fast\_2\_timer/e\_po3030k.h, 110  
   slow\_3\_timer/e\_po3030k.h, 128  
 e\_po3030k\_set\_awb\_ae  
   e\_po3030k\_registers.c, 145  
   e\_registers.c, 199  
   fast\_2\_timer/e\_po3030k.h, 110  
   slow\_3\_timer/e\_po3030k.h, 128  
 e\_po3030k\_set\_awb\_ae\_tol  
   e\_po3030k\_registers.c, 146  
   e\_registers.c, 199  
   fast\_2\_timer/e\_po3030k.h, 110  
   slow\_3\_timer/e\_po3030k.h, 129  
 e\_po3030k\_set\_bias  
   e\_po3030k\_registers.c, 146  
   e\_registers.c, 200  
   fast\_2\_timer/e\_po3030k.h, 111  
   slow\_3\_timer/e\_po3030k.h, 129  
 e\_po3030k\_set\_brigh\_contr  
   e\_po3030k\_registers.c, 146  
   e\_registers.c, 200  
   fast\_2\_timer/e\_po3030k.h, 111  
   slow\_3\_timer/e\_po3030k.h, 129  
 e\_po3030k\_set\_cb\_cr\_gain  
   e\_po3030k\_registers.c, 147  
   e\_registers.c, 200  
   fast\_2\_timer/e\_po3030k.h, 111  
   slow\_3\_timer/e\_po3030k.h, 130  
 e\_po3030k\_set\_color\_gain  
   e\_po3030k\_registers.c, 147  
   e\_registers.c, 200  
   fast\_2\_timer/e\_po3030k.h, 111  
   slow\_3\_timer/e\_po3030k.h, 130  
 e\_po3030k\_set\_color\_matrix  
   fast\_2\_timer/e\_po3030k.h, 112  
   slow\_3\_timer/e\_po3030k.h, 130  
 e\_po3030k\_set\_color\_mode  
   e\_po3030k\_registers.c, 147  
   e\_registers.c, 201  
   fast\_2\_timer/e\_po3030k.h, 112  
   slow\_3\_timer/e\_po3030k.h, 130  
 e\_po3030k\_set\_edge\_prop  
   e\_po3030k\_registers.c, 148  
   e\_registers.c, 201  
   fast\_2\_timer/e\_po3030k.h, 112  
   slow\_3\_timer/e\_po3030k.h, 131



- e\_po3030k\_set\_exposure
  - e\_po3030k\_registers.c, 148
  - e\_registers.c, 202
  - fast\_2\_timer/e\_po3030k.h, 113
  - slow\_3\_timer/e\_po3030k.h, 131
- e\_po3030k\_set\_flicker\_detection
  - e\_po3030k\_registers.c, 149
  - e\_registers.c, 202
  - fast\_2\_timer/e\_po3030k.h, 113
  - slow\_3\_timer/e\_po3030k.h, 131
- e\_po3030k\_set\_flicker\_man\_set
  - e\_po3030k\_registers.c, 149
  - e\_registers.c, 202
  - fast\_2\_timer/e\_po3030k.h, 113
  - slow\_3\_timer/e\_po3030k.h, 132
- e\_po3030k\_set\_flicker\_mode
  - e\_po3030k\_registers.c, 149
  - e\_registers.c, 203
  - fast\_2\_timer/e\_po3030k.h, 114
  - slow\_3\_timer/e\_po3030k.h, 132
- e\_po3030k\_set\_gamma\_coef
  - e\_po3030k\_registers.c, 150
  - e\_registers.c, 203
  - fast\_2\_timer/e\_po3030k.h, 114
  - slow\_3\_timer/e\_po3030k.h, 133
- e\_po3030k\_set\_integr\_time
  - e\_po3030k\_registers.c, 150
  - e\_registers.c, 204
  - fast\_2\_timer/e\_po3030k.h, 115
  - slow\_3\_timer/e\_po3030k.h, 133
- e\_po3030k\_set\_lens\_gain
  - e\_po3030k\_registers.c, 150
  - e\_registers.c, 204
  - fast\_2\_timer/e\_po3030k.h, 115
  - slow\_3\_timer/e\_po3030k.h, 133
- e\_po3030k\_set\_max\_min\_awb
  - e\_po3030k\_registers.c, 151
  - e\_registers.c, 204
  - fast\_2\_timer/e\_po3030k.h, 115
  - slow\_3\_timer/e\_po3030k.h, 134
- e\_po3030k\_set\_max\_min\_exp
  - e\_po3030k\_registers.c, 151
  - e\_registers.c, 205
  - fast\_2\_timer/e\_po3030k.h, 116
  - slow\_3\_timer/e\_po3030k.h, 134
- e\_po3030k\_set\_mirror
  - e\_po3030k\_registers.c, 151
  - e\_registers.c, 205
  - fast\_2\_timer/e\_po3030k.h, 116
  - slow\_3\_timer/e\_po3030k.h, 134
- e\_po3030k\_set\_ref\_exposure
  - e\_po3030k\_registers.c, 152
  - e\_registers.c, 205
  - fast\_2\_timer/e\_po3030k.h, 116
  - slow\_3\_timer/e\_po3030k.h, 135
- e\_po3030k\_set\_register
  - e\_po3030k\_registers.c, 152
  - e\_registers.c, 205
  - fast\_2\_timer/e\_po3030k.h, 116
  - slow\_3\_timer/e\_po3030k.h, 135
- fast\_2\_timer/e\_po3030k.h, 116
- slow\_3\_timer/e\_po3030k.h, 135
- e\_po3030k\_set\_sampling\_mode
  - e\_po3030k\_registers.c, 152
  - e\_registers.c, 206
  - fast\_2\_timer/e\_po3030k.h, 117
  - slow\_3\_timer/e\_po3030k.h, 135
- e\_po3030k\_set\_sepia
  - e\_po3030k\_registers.c, 153
  - e\_registers.c, 206
  - fast\_2\_timer/e\_po3030k.h, 117
  - slow\_3\_timer/e\_po3030k.h, 136
- e\_po3030k\_set\_sepia\_tone
  - e\_po3030k\_registers.c, 153
  - e\_registers.c, 207
  - fast\_2\_timer/e\_po3030k.h, 118
  - slow\_3\_timer/e\_po3030k.h, 136
- e\_po3030k\_set\_speed
  - e\_po3030k\_registers.c, 153
  - e\_registers.c, 207
  - fast\_2\_timer/e\_po3030k.h, 118
  - slow\_3\_timer/e\_po3030k.h, 136
- e\_po3030k\_set\_vsync
  - e\_po3030k\_registers.c, 154
  - e\_registers.c, 207
  - fast\_2\_timer/e\_po3030k.h, 118
  - slow\_3\_timer/e\_po3030k.h, 137
- e\_po3030k\_set\_weight\_win
  - e\_po3030k\_registers.c, 154
  - e\_registers.c, 208
  - fast\_2\_timer/e\_po3030k.h, 119
  - slow\_3\_timer/e\_po3030k.h, 137
- e\_po3030k\_set\_ww
  - e\_po3030k\_registers.c, 155
  - e\_registers.c, 208
  - fast\_2\_timer/e\_po3030k.h, 119
  - slow\_3\_timer/e\_po3030k.h, 138
- e\_po3030k\_set\_wx
  - e\_po3030k\_registers.c, 155
  - e\_registers.c, 208
  - fast\_2\_timer/e\_po3030k.h, 119
  - slow\_3\_timer/e\_po3030k.h, 138
- e\_po3030k\_set\_wy
  - e\_po3030k\_registers.c, 155
  - e\_registers.c, 209
  - fast\_2\_timer/e\_po3030k.h, 120
  - slow\_3\_timer/e\_po3030k.h, 138
- e\_po3030k\_sync\_register\_array
  - e\_po3030k\_registers.c, 156
  - e\_registers.c, 210
  - fast\_2\_timer/e\_po3030k.h, 120
  - slow\_3\_timer/e\_po3030k.h, 139
- e\_po3030k\_write\_cam\_registers
  - e\_po3030k\_registers.c, 156
  - e\_registers.c, 210
  - fast\_2\_timer/e\_po3030k.h, 122
  - slow\_3\_timer/e\_po3030k.h, 139
- e\_po3030k\_write\_gamma\_coef

- e\_po3030k\_registers.c, 157
- e\_registers.c, 210
- fast\_2\_timer/e\_po3030k.h, 122
- slow\_3\_timer/e\_po3030k.h, 139
- e\_po6030k.h
  - BANK\_A, 158
  - BANK\_B, 158
  - BANK\_C, 158
  - BANK\_D, 158
  - BAYER\_CLOCK\_1, 158
  - BAYER\_CLOCK\_2, 158
  - BAYER\_CLOCK\_4, 158
  - BAYER\_CLOCK\_8, 158
  - E\_PO6030K\_SKETCH\_BW, 158
  - E\_PO6030K\_SKETCH\_COLOR, 159
  - e\_po6030k\_config\_cam, 159
  - e\_po6030k\_get\_bytes\_per\_pixel, 159
  - e\_po6030k\_set\_awb\_ae, 161
  - e\_po6030k\_set\_bank, 161
  - e\_po6030k\_set\_bayer\_clkdiv, 161
  - e\_po6030k\_set\_color\_mode, 161
  - e\_po6030k\_set\_exposure, 161
  - e\_po6030k\_set\_mirror, 163
  - e\_po6030k\_set\_mode, 163
  - e\_po6030k\_set\_pclkdiv, 163
  - e\_po6030k\_set\_rgb\_gain, 163
  - e\_po6030k\_set\_sketch\_mode, 163
  - e\_po6030k\_set\_speed, 158
  - e\_po6030k\_set\_vsync, 164
  - e\_po6030k\_set\_wx, 164
  - e\_po6030k\_set\_wy, 164
  - e\_po6030k\_write\_register, 165
  - PO\_6030\_MODE\_QQVGA, 159
  - PO\_6030\_MODE\_QVGA, 159
  - PO\_6030\_MODE\_VGA, 159
  - PO\_6030\_SPEED\_1, 159
  - PO\_6030\_SPEED\_2, 159
  - PO\_6030\_SPEED\_4, 159
  - PO\_6030\_SPEED\_8, 159
- e\_po6030k\_config\_cam
  - e\_calc\_po6030k.c, 101
  - e\_po6030k.h, 159
- e\_po6030k\_get\_bytes\_per\_pixel
  - e\_calc\_po6030k.c, 101
  - e\_po6030k.h, 159
- e\_po6030k\_read\_register
  - e\_po6030k\_registers.c, 166
- e\_po6030k\_registers.c
  - ARRAY\_ORIGINE\_X, 166
  - ARRAY\_ORIGINE\_Y, 166
  - BANK\_REGISTER, 166
  - DEVICE\_ID, 166
  - e\_po6030k\_read\_register, 166
  - e\_po6030k\_set\_awb\_ae, 166
  - e\_po6030k\_set\_bank, 167
  - e\_po6030k\_set\_bayer\_clkdiv, 167
  - e\_po6030k\_set\_exposure, 167
  - e\_po6030k\_set\_mirror, 167
  - e\_po6030k\_set\_mode, 168
  - e\_po6030k\_set\_pclkdiv, 168
  - e\_po6030k\_set\_rgb\_gain, 168
  - e\_po6030k\_set\_sampl\_color, 168
  - e\_po6030k\_set\_sampl\_gray, 168
  - e\_po6030k\_set\_sketch\_mode, 168
  - e\_po6030k\_set\_speed, 158
  - e\_po6030k\_set\_vsync, 164
  - e\_po6030k\_set\_wx, 169
  - e\_po6030k\_set\_wy, 169
  - e\_po6030k\_write\_register, 169
  - MCLK\_P\_NS, 166
  - MCLK, 166
- e\_po6030k\_set\_awb\_ae
  - e\_po6030k.h, 161
  - e\_po6030k\_registers.c, 166
- e\_po6030k\_set\_bank
  - e\_po6030k.h, 161
  - e\_po6030k\_registers.c, 167
- e\_po6030k\_set\_bayer\_clkdiv
  - e\_po6030k.h, 161
  - e\_po6030k\_registers.c, 167
- e\_po6030k\_set\_color\_mode
  - e\_po6030k.h, 161
- e\_po6030k\_set\_exposure
  - e\_po6030k.h, 161
  - e\_po6030k\_registers.c, 167
- e\_po6030k\_set\_mirror
  - e\_po6030k.h, 163
  - e\_po6030k\_registers.c, 167
- e\_po6030k\_set\_mode
  - e\_po6030k.h, 163
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_pclkdiv
  - e\_po6030k.h, 163
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_rgb\_gain
  - e\_po6030k.h, 163
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_sampl\_color
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_sampl\_gray
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_sketch\_mode
  - e\_po6030k.h, 163
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_speed
  - e\_po6030k.h, 158
- e\_po6030k\_set\_vsync
  - e\_po6030k.h, 164
  - e\_po6030k\_registers.c, 168
- e\_po6030k\_set\_wx
  - e\_po6030k.h, 164
  - e\_po6030k\_registers.c, 169
- e\_po6030k\_set\_wy
  - e\_po6030k.h, 164
  - e\_po6030k\_registers.c, 169
- e\_po6030k\_write\_register
  - e\_po6030k.h, 165

- e\_po6030k\_registers.c, 169
- e\_po8030d.h
  - BANK\_A, 171
  - BANK\_B, 171
  - BANK\_C, 171
  - BANK\_D, 171
  - E\_PO8030D\_SKETCH\_BW, 171
  - E\_PO8030D\_SKETCH\_COLOR, 171
  - e\_po8030d\_config\_cam, 171
  - e\_po8030d\_get\_bytes\_per\_pixel, 172
  - e\_po8030d\_set\_awb\_ae, 172
  - e\_po8030d\_set\_bank, 172
  - e\_po8030d\_set\_bayer\_clkdiv, 173
  - e\_po8030d\_set\_brightness, 173
  - e\_po8030d\_set\_color\_mode, 173
  - e\_po8030d\_set\_exposure, 173
  - e\_po8030d\_set\_mirror, 173
  - e\_po8030d\_set\_mode, 173
  - e\_po8030d\_set\_pclkdiv, 174
  - e\_po8030d\_set\_rgb\_gain, 174
  - e\_po8030d\_set\_sketch\_mode, 174
  - e\_po8030d\_set\_speed, 171
  - e\_po8030d\_set\_vsync, 174
  - e\_po8030d\_set\_wx, 175
  - e\_po8030d\_set\_wy, 175
  - e\_po8030d\_write\_register, 175
  - init\_po8030, 176
  - PO\_8030\_BAYER\_CLOCK\_1, 171
  - PO\_8030\_BAYER\_CLOCK\_2, 171
  - PO\_8030\_BAYER\_CLOCK\_4, 171
  - PO\_8030\_BAYER\_CLOCK\_8, 171
  - PO\_8030\_MODE\_QQVGA, 171
  - PO\_8030\_MODE\_QVGA, 171
  - PO\_8030\_MODE\_VGA, 171
  - PO\_8030\_SPEED\_1, 171
  - PO\_8030\_SPEED\_2, 171
  - PO\_8030\_SPEED\_4, 171
  - PO\_8030\_SPEED\_8, 171
  - testWriteReg, 176
- e\_po8030d\_config\_cam
  - e\_calc\_po8030d.c, 102
  - e\_po8030d.h, 171
- e\_po8030d\_get\_bytes\_per\_pixel
  - e\_calc\_po8030d.c, 103
  - e\_po8030d.h, 172
- e\_po8030d\_read\_register
  - e\_po8030d\_registers.c, 177
- e\_po8030d\_registers.c
  - ARRAY\_ORIGINE\_X, 177
  - ARRAY\_ORIGINE\_Y, 177
  - BANK\_REGISTER, 177
  - DEVICE\_ID, 177
  - e\_po8030d\_read\_register, 177
  - e\_po8030d\_set\_awb\_ae, 177
  - e\_po8030d\_set\_bank, 178
  - e\_po8030d\_set\_bayer\_clkdiv, 178
  - e\_po8030d\_set\_brightness, 178
  - e\_po8030d\_set\_exposure, 178
  - e\_po8030d\_set\_mirror, 179
  - e\_po8030d\_set\_mode, 179
  - e\_po8030d\_set\_pclkdiv, 179
  - e\_po8030d\_set\_rgb\_gain, 179
  - e\_po8030d\_set\_sampl\_color, 179
  - e\_po8030d\_set\_sampl\_gray, 179
  - e\_po8030d\_set\_sketch\_mode, 179
  - e\_po8030d\_set\_vsync, 180
  - e\_po8030d\_set\_wx, 180
  - e\_po8030d\_set\_wy, 180
  - e\_po8030d\_write\_register, 181
  - enablePO8030, 181
  - init\_po8030, 181
  - MCLK\_P\_NS, 177
  - MCLK, 177
  - testWriteReg, 181
- e\_po8030d\_set\_awb\_ae
  - e\_po8030d.h, 172
  - e\_po8030d\_registers.c, 177
- e\_po8030d\_set\_bank
  - e\_po8030d.h, 172
  - e\_po8030d\_registers.c, 178
- e\_po8030d\_set\_bayer\_clkdiv
  - e\_po8030d.h, 173
  - e\_po8030d\_registers.c, 178
- e\_po8030d\_set\_brightness
  - e\_po8030d.h, 173
  - e\_po8030d\_registers.c, 178
- e\_po8030d\_set\_color\_mode
  - e\_po8030d.h, 173
- e\_po8030d\_set\_exposure
  - e\_po8030d.h, 173
  - e\_po8030d\_registers.c, 178
- e\_po8030d\_set\_mirror
  - e\_po8030d.h, 173
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_mode
  - e\_po8030d.h, 173
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_pclkdiv
  - e\_po8030d.h, 174
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_rgb\_gain
  - e\_po8030d.h, 174
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_sampl\_color
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_sampl\_gray
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_sketch\_mode
  - e\_po8030d.h, 174
  - e\_po8030d\_registers.c, 179
- e\_po8030d\_set\_speed
  - e\_po8030d.h, 171
- e\_po8030d\_set\_vsync
  - e\_po8030d.h, 174
  - e\_po8030d\_registers.c, 180
- e\_po8030d\_set\_wx

- e\_po8030d.h, 175
- e\_po8030d\_registers.c, 180
- e\_po8030d\_set\_wy
  - e\_po8030d.h, 175
  - e\_po8030d\_registers.c, 180
- e\_po8030d\_write\_register
  - e\_po8030d.h, 175
  - e\_po8030d\_registers.c, 181
- e\_poxxxx.h
  - ARRAY\_HEIGHT, 182
  - ARRAY\_WIDTH, 182
  - e\_poxxxx\_apply\_timer\_config, 182
  - e\_poxxxx\_config\_cam, 183
  - e\_poxxxx\_get\_orientation, 183
  - e\_poxxxx\_init\_cam, 183
  - e\_poxxxx\_is\_img\_ready, 183
  - e\_poxxxx\_launch\_capture, 183
  - e\_poxxxx\_set\_awb\_ae, 183
  - e\_poxxxx\_set\_exposure, 185
  - e\_poxxxx\_set\_mirror, 185
  - e\_poxxxx\_set\_rgb\_gain, 185
  - e\_poxxxx\_write\_cam\_registers, 185
  - GREY\_SCALE\_MODE, 182
  - POXXXX\_FULL, 182
  - RGB\_565\_MODE, 182
  - YUV\_MODE, 182
- e\_poxxxx\_apply\_timer\_config
  - e\_poxxxx.h, 182
  - fast\_2\_timer/e\_timers.c, 186
- e\_poxxxx\_config\_cam
  - e\_common.c, 104
  - e\_poxxxx.h, 183
- e\_poxxxx\_get\_orientation
  - e\_common.c, 104
  - e\_poxxxx.h, 183
- e\_poxxxx\_init\_cam
  - e\_common.c, 104
  - e\_poxxxx.h, 183
- e\_poxxxx\_is\_img\_ready
  - e\_poxxxx.h, 183
  - fast\_2\_timer/e\_timers.c, 187
- e\_poxxxx\_launch\_capture
  - e\_poxxxx.h, 183
  - fast\_2\_timer/e\_timers.c, 187
- e\_poxxxx\_set\_awb\_ae
  - e\_common.c, 104
  - e\_poxxxx.h, 183
- e\_poxxxx\_set\_exposure
  - e\_common.c, 104
  - e\_poxxxx.h, 185
- e\_poxxxx\_set\_mirror
  - e\_common.c, 105
  - e\_poxxxx.h, 185
- e\_poxxxx\_set\_rgb\_gain
  - e\_common.c, 105
  - e\_poxxxx.h, 185
- e\_poxxxx\_write\_cam\_registers
  - e\_common.c, 105
- e\_poxxxx.h, 185
- e\_prox.c
  - \_\_attribute\_\_, 67
  - ambient\_and\_reflected\_ir, 69
  - ambient\_ir, 69
  - e\_get\_ambient\_light, 67
  - e\_get\_prox, 68
  - e\_init\_prox, 68
  - e\_stop\_prox, 68
  - init\_tmr1, 68
  - reflected\_ir, 69
- e\_prox.h
  - e\_get\_ambient\_light, 73
  - e\_get\_prox, 73
  - e\_init\_prox, 74
  - e\_stop\_prox, 74
- e\_prox\_timer2.c
  - \_\_attribute\_\_, 78
  - ambient\_and\_reflected\_ir, 79
  - ambient\_ir, 79
  - e\_get\_ambient\_light, 78
  - e\_get\_prox, 78
  - e\_init\_prox, 79
  - e\_stop\_prox, 79
  - init\_tmr2, 79
  - reflected\_ir, 79
- e\_read\_acc
  - e\_acc.c, 42
  - e\_acc.h, 47
- e\_read\_acc\_spheric
  - e\_acc.c, 42
  - e\_acc.h, 48
- e\_read\_acc\_x
  - e\_acc.c, 42
  - e\_acc.h, 48
- e\_read\_acc\_xyz
  - e\_acc.c, 42
  - e\_acc.h, 48
- e\_read\_acc\_y
  - e\_acc.c, 43
  - e\_acc.h, 48
- e\_read\_acc\_z
  - e\_acc.c, 43
  - e\_acc.h, 48
- e\_read\_ad
  - e\_ad\_conv.c, 52
  - e\_ad\_conv.h, 56
- e\_read\_inclination
  - e\_acc.c, 43
  - e\_acc.h, 49
- e\_read\_orientation
  - e\_acc.c, 43
  - e\_acc.h, 49
- e\_read\_remote\_control
  - e\_remote\_control.c, 311
  - e\_remote\_control.h, 315
- e\_receive\_char\_from\_matlab
  - matlab.c, 245

- matlab.h, [247](#)
- e\_receive\_int\_from\_matlab
  - matlab.c, [246](#)
  - matlab.h, [247](#)
- e\_registers.c
  - ADCOFF\_BASE, [196](#)
  - AESPEED\_BASE, [196](#)
  - ARRAY\_ORIGINE\_X, [196](#)
  - ARRAY\_ORIGINE\_Y, [196](#)
  - AWBAEENABLE\_BASE, [196](#)
  - AWVAETOL\_BASE, [196](#)
  - BASE\_D1, [196](#)
  - BASE\_D2, [196](#)
  - BASE\_D3, [196](#)
  - BASE\_D4, [196](#)
  - BIAS\_BASE, [196](#)
  - BRICTR\_BASE, [196](#)
  - CBCRGAIN\_BASE, [196](#)
  - COLGAIN\_BASE, [196](#)
  - COLOR\_COEF\_BASE, [196](#)
  - COLOR\_M\_ADDR, [196](#)
  - cam\_reg, [211](#)
  - DEVICE\_ID, [196](#)
  - e\_po3030k\_SetColorMatrix, [209](#)
  - e\_po3030k\_get\_register, [198](#)
  - e\_po3030k\_read\_cam\_registers, [198](#)
  - e\_po3030k\_set\_adc\_offset, [198](#)
  - e\_po3030k\_set\_ae\_speed, [199](#)
  - e\_po3030k\_set\_awb\_ae, [199](#)
  - e\_po3030k\_set\_awb\_ae\_tol, [199](#)
  - e\_po3030k\_set\_bias, [200](#)
  - e\_po3030k\_set\_brigh\_contr, [200](#)
  - e\_po3030k\_set\_cb\_cr\_gain, [200](#)
  - e\_po3030k\_set\_color\_gain, [200](#)
  - e\_po3030k\_set\_color\_mode, [201](#)
  - e\_po3030k\_set\_edge\_prop, [201](#)
  - e\_po3030k\_set\_exposure, [202](#)
  - e\_po3030k\_set\_flicker\_detection, [202](#)
  - e\_po3030k\_set\_flicker\_man\_set, [202](#)
  - e\_po3030k\_set\_flicker\_mode, [203](#)
  - e\_po3030k\_set\_gamma\_coef, [203](#)
  - e\_po3030k\_set\_integr\_time, [204](#)
  - e\_po3030k\_set\_lens\_gain, [204](#)
  - e\_po3030k\_set\_max\_min\_awb, [204](#)
  - e\_po3030k\_set\_max\_min\_exp, [205](#)
  - e\_po3030k\_set\_mirror, [205](#)
  - e\_po3030k\_set\_ref\_exposure, [205](#)
  - e\_po3030k\_set\_register, [205](#)
  - e\_po3030k\_set\_sampling\_mode, [206](#)
  - e\_po3030k\_set\_sepia, [206](#)
  - e\_po3030k\_set\_sepia\_tone, [207](#)
  - e\_po3030k\_set\_speed, [207](#)
  - e\_po3030k\_set\_vsync, [207](#)
  - e\_po3030k\_set\_weight\_win, [208](#)
  - e\_po3030k\_set\_ww, [208](#)
  - e\_po3030k\_set\_wx, [208](#)
  - e\_po3030k\_set\_wy, [209](#)
  - e\_po3030k\_sync\_register\_array, [210](#)
  - e\_po3030k\_write\_cam\_registers, [210](#)
  - e\_po3030k\_write\_gamma\_coef, [210](#)
  - EDGE\_BASE, [196](#)
  - EXPOSURE\_BASE, [196](#)
  - FLICKM\_BASE, [196](#)
  - FLICKP\_BASE, [196](#)
  - FRAME\_HEIGHT, [196](#)
  - FRAME\_WIDTH, [196](#)
  - GAMMA\_BASE, [197](#)
  - GAMMASELCOL\_BASE, [197](#)
  - INTEGR\_BASE, [197](#)
  - LENSG\_BASE, [197](#)
  - MCLK\_P\_NS, [197](#)
  - MCLK, [197](#)
  - MINMAXAWB\_BASE, [197](#)
  - MINMAXEXP\_BASE, [197](#)
  - MIRROR\_BASE, [197](#)
  - MODE\_GRAYSCALE, [197](#)
  - MODE\_R5G6B5, [197](#)
  - MODE\_YUV, [197](#)
  - NB\_REGISTERS, [197](#)
  - po3030k\_get\_pixelclock, [210](#)
  - REFREPO\_BASE, [197](#)
  - SAMPLING\_ADDR, [197](#)
  - SEPIA\_BASE, [197](#)
  - SEPIATONE\_BASE, [197](#)
  - SPEED\_ADDR, [197](#)
  - VSYNC COL\_BASE, [197](#)
  - VSYNCSTART\_BASE, [197](#)
  - VSYNCSTOP\_BASE, [197](#)
  - WEIGHWIN\_BASE, [197](#)
  - WINDOW\_X1\_BASE, [197](#)
  - WINDOW\_X2\_BASE, [198](#)
  - WINDOW\_Y1\_BASE, [198](#)
  - WINDOW\_Y2\_BASE, [198](#)
  - WW\_BASE, [198](#)
  - e\_remote\_control.c
    - \_\_attribute\_\_, [311](#)
    - address, [312](#)
    - address\_temp, [312](#)
    - check, [312](#)
    - check\_temp, [312](#)
    - data, [312](#)
    - data\_temp, [312](#)
    - e\_get\_address, [311](#)
    - e\_get\_check, [311](#)
    - e\_get\_data, [311](#)
    - e\_init\_remote\_control, [311](#)
    - e\_read\_remote\_control, [311](#)
  - e\_remote\_control.h
    - BOTTOMI, [314](#)
    - BOTTOMR, [314](#)
    - CHAN\_DOWN, [314](#)
    - CHAN\_UP, [314](#)
    - e\_get\_address, [314](#)
    - e\_get\_check, [314](#)
    - e\_get\_data, [314](#)
    - e\_init\_remote\_control, [314](#)

- e\_read\_remote\_control, 315
- I\_II, 314
- MUTE, 314
- OUT\_AUX\_1, 314
- STANDBY, 314
- VOL\_DOWN, 314
- VOL\_UP, 314
- e\_reset\_agenda
  - e\_agenda.c, 251
  - e\_agenda.h, 256
  - e\_agenda\_fast.c, 318
  - e\_agenda\_fast.h, 323
- e\_restart\_agenda
  - e\_agenda.c, 251
  - e\_agenda.h, 256
  - e\_agenda\_fast.c, 318
  - e\_agenda\_fast.h, 323
- e\_send\_char\_to\_matlab
  - matlab.c, 246
  - matlab.h, 248
- e\_send\_int\_to\_matlab
  - matlab.c, 246
  - matlab.h, 248
- e\_send\_uart1\_char
  - e\_uart\_char.h, 340
- e\_send\_uart2\_char
  - e\_uart\_char.h, 341
- e\_sensext.c
  - e\_get\_sensext\_wait, 217
  - e\_init\_sensext, 217
  - e\_sensext\_process, 217
  - e\_start\_sensext\_wait, 217
  - e\_stop\_sensext\_wait, 217
  - sensext\_wait, 217
- e\_sensext.h
  - e\_get\_sensext\_wait, 218
  - e\_init\_sensext, 218
  - e\_sensext\_process, 218
  - e\_start\_sensext\_wait, 218
  - e\_stop\_sensext\_wait, 218
  - I2C\_ADDR\_CMPS03, 218
  - I2C\_ADDR\_SENSEXT, 218
  - I2C\_ADDR\_SRF08, 218
  - I2C\_ADDR\_SRF10, 218
  - I2C\_ADDR\_SRF235, 218
- e\_sensext\_process
  - e\_sensext.c, 217
  - e\_sensext.h, 218
- e\_set\_agenda\_cycle
  - e\_agenda.c, 251
  - e\_agenda.h, 257
  - e\_agenda\_fast.c, 318
  - e\_agenda\_fast.h, 324
- e\_set\_blinking\_cycle
  - advance\_one\_timer/e\_led.c, 262
  - advance\_one\_timer/fast\_agenda/e\_led.c, 267
- e\_set\_body\_led
  - advance\_one\_timer/e\_led.c, 262
- advance\_one\_timer/e\_led.h, 275
- advance\_one\_timer/fast\_agenda/e\_led.c, 267
- advance\_one\_timer/fast\_agenda/e\_led.h, 281
- e\_led.c, 269
- e\_led.h, 284
- e\_set\_front\_led
  - advance\_one\_timer/e\_led.c, 262
  - advance\_one\_timer/e\_led.h, 275
  - advance\_one\_timer/fast\_agenda/e\_led.c, 267
  - advance\_one\_timer/fast\_agenda/e\_led.h, 281
  - e\_led.c, 271
  - e\_led.h, 284
- e\_set\_gain\_devantech
  - e\_devantech.c, 215
  - e\_devantech.h, 216
- e\_set\_led
  - advance\_one\_timer/e\_led.c, 262
  - advance\_one\_timer/e\_led.h, 275
  - advance\_one\_timer/fast\_agenda/e\_led.c, 268
  - advance\_one\_timer/fast\_agenda/e\_led.h, 282
  - e\_led.c, 271
  - e\_led.h, 284
- e\_set\_motor\_speed
  - e\_agenda\_fast.c, 318
  - e\_agenda\_fast.h, 324
- e\_set\_range\_devantech
  - e\_devantech.c, 215
  - e\_devantech.h, 216
- e\_set\_sharp\_led
  - e\_sharp.c, 219
  - e\_sharp.h, 220
- e\_set\_speed
  - advance\_one\_timer/e\_motors.c, 287
  - advance\_one\_timer/e\_motors.h, 298
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 291
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 302
- e\_set\_speed\_left
  - advance\_one\_timer/e\_motors.c, 287
  - advance\_one\_timer/e\_motors.h, 298
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 291
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 303
  - e\_motors.c, 295
  - e\_motors.h, 307
  - e\_motors\_timer3.c, 336
- e\_set\_speed\_right
  - advance\_one\_timer/e\_motors.c, 287
  - advance\_one\_timer/e\_motors.h, 299
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 291
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 304
  - e\_motors.c, 295
  - e\_motors.h, 308
  - e\_motors\_timer3.c, 336
- e\_set\_steps\_left
  - advance\_one\_timer/e\_motors.c, 288
  - advance\_one\_timer/e\_motors.h, 300
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 305
  - e\_motors.c, 295

- e\_motors.h, 309
- e\_motors\_timer3.c, 337
- e\_set\_steps\_right
  - advance\_one\_timer/e\_motors.c, 288
  - advance\_one\_timer/e\_motors.h, 301
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
  - advance\_one\_timer/fast\_agenda/e\_motors.h, 305
  - e\_motors.c, 296
  - e\_motors.h, 309
  - e\_motors\_timer3.c, 337
- e\_set\_timer\_speed
  - e\_agenda\_fast.c, 318
- e\_sharp.c
  - e\_get\_dist\_sharp, 219
  - e\_init\_sharp, 219
  - e\_set\_sharp\_led, 219
  - e\_sharp\_led\_clear, 219
  - e\_sharp\_off, 219
  - e\_sharp\_on, 219
- e\_sharp.h
  - e\_get\_dist\_sharp, 220
  - e\_init\_sharp, 220
  - e\_set\_sharp\_led, 220
  - e\_sharp\_led\_clear, 220
  - e\_sharp\_off, 220
  - e\_sharp\_on, 220
  - SHARP\_LED1, 220
  - SHARP\_LED1\_DIR, 220
  - SHARP\_LED2, 220
  - SHARP\_LED2\_DIR, 220
  - SHARP\_LED3, 220
  - SHARP\_LED3\_DIR, 220
  - SHARP\_LED4, 220
  - SHARP\_LED4\_DIR, 220
  - SHARP\_LED5, 220
  - SHARP\_LED5\_DIR, 220
  - SHARP\_VIN\_DIR, 220
  - SHARP\_VIN, 220
  - SHARP, 220
- e\_sharp\_led\_clear
  - e\_sharp.c, 219
  - e\_sharp.h, 220
- e\_sharp\_off
  - e\_sharp.c, 219
  - e\_sharp.h, 220
- e\_sharp\_on
  - e\_sharp.c, 219
  - e\_sharp.h, 220
- e\_sound.c
  - e\_close\_sound, 211
  - e\_init\_sound, 211
  - e\_play\_sound, 211
- e\_sound.h
  - e\_close\_sound, 212
  - e\_init\_codec\_slave, 212
  - e\_init\_dci\_master, 213
  - e\_init\_sound, 213
  - e\_play\_sound, 213
  - e\_sub\_dci\_kickoff, 214
- e\_start\_agendas\_processing
  - e\_agenda.c, 252
  - e\_agenda.h, 257
  - e\_agenda\_fast.c, 318
  - e\_agenda\_fast.h, 324
- e\_start\_led\_blinking
  - advance\_one\_timer/e\_led.c, 263
  - advance\_one\_timer/e\_led.h, 277
  - advance\_one\_timer/fast\_agenda/e\_led.c, 268
  - advance\_one\_timer/fast\_agenda/e\_led.h, 282
- e\_start\_sensex\_t\_wait
  - e\_sensex.c, 217
  - e\_sensex.h, 218
- e\_start\_timer\_processing
  - e\_agenda\_fast.c, 318
  - e\_agenda\_fast.h, 324
- e\_stop\_led\_blinking
  - advance\_one\_timer/e\_led.c, 263
  - advance\_one\_timer/e\_led.h, 277
  - advance\_one\_timer/fast\_agenda/e\_led.c, 268
  - advance\_one\_timer/fast\_agenda/e\_led.h, 283
- e\_stop\_prox
  - e\_prox.c, 68
  - e\_prox.h, 74
  - e\_prox\_timer2.c, 79
- e\_stop\_sensex\_t\_wait
  - e\_sensex.c, 217
  - e\_sensex.h, 218
- e\_sub\_dci\_kickoff
  - e\_sound.h, 214
- e\_subtract\_mean
  - e\_fft\_utilities.h, 228
- e\_twiddle\_factors.c
  - \_\_attribute\_\_, 229
- e\_uart1\_int\_clr\_addr
  - e\_uart\_char.h, 341
- e\_uart1\_int\_clr\_mask
  - e\_uart\_char.h, 341
- e\_uart1\_sending
  - e\_uart\_char.h, 341
- e\_uart2\_int\_clr\_addr
  - e\_uart\_char.h, 341
- e\_uart2\_int\_clr\_mask
  - e\_uart\_char.h, 341
- e\_uart2\_sending
  - e\_uart\_char.h, 341
- e\_uart\_char.h
  - BAUD115200, 339
  - BAUD230400, 339
  - BAUD460800, 339
  - BAUD921600, 339
  - e\_getchar\_uart1, 339
  - e\_getchar\_uart2, 340
  - e\_init\_uart1, 340
  - e\_init\_uart2, 340
  - e\_ischar\_uart1, 340
  - e\_ischar\_uart2, 340

- e\_send\_uart1\_char, 340
- e\_send\_uart2\_char, 341
- e\_uart1\_int\_clr\_addr, 341
- e\_uart1\_int\_clr\_mask, 341
- e\_uart1\_sending, 341
- e\_uart2\_int\_clr\_addr, 341
- e\_uart2\_int\_clr\_mask, 341
- e\_uart2\_sending, 341
- EDGE\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- ERROR
  - e\_i2C\_master\_module.h, 234
- EXIT\_OK
  - e\_agenda.c, 249
  - e\_agenda\_fast.c, 316
- EXPOSURE\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- elseif
  - EpuckGetData.m, 243
- enablePO8030
  - e\_po8030d\_registers.c, 181
- EpuckFlush.m
  - EpuckPort, 242
  - flush, 242
  - flushinput, 242
  - flushoutput, 242
  - function, 242
- EpuckGetData.m
  - continue, 243
  - data, 243
  - elseif, 243
  - function, 243
  - i, 243
  - if, 243
  - receivedFormat, 243
  - return, 243
  - while, 243
- EpuckPort
  - CloseEpuck.m, 241
  - EpuckFlush.m, 242
  - OpenEpuck.m, 245
- EpuckSendData.m
  - function, 244
  - fwrite, 244
  - if, 244
  - int16, 244
  - return, 244
  - size, 244
- Error
  - OpenEpuck.m, 245
- FALSE
  - e\_epuck\_ports.h, 329
- FCY
  - e\_epuck\_ports.h, 329
- FFT\_BLOCK\_LENGTH
  - e\_fft.h, 227
- FFT, 23
- FILTER\_SIZE
  - e\_acc.h, 46
- FIRSTDATA\_IN\_PACKET\_OFFSET
  - ComModule.c, 222
- FLICKM\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- FLICKP\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- FOSC
  - e\_epuck\_ports.h, 329
- FRAME\_HEIGHT
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- FRAME\_WIDTH
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 196
- FRONT\_LED\_DIR
  - e\_epuck\_ports.h, 329
- FRONT\_LED
  - e\_epuck\_ports.h, 329
- fast\_2\_timer/e\_po3030k.h
  - e\_po3030k\_config\_cam, 107
  - e\_po3030k\_get\_bytes\_per\_pixel, 108
  - e\_po3030k\_get\_register, 109
  - e\_po3030k\_init\_cam, 109
  - e\_po3030k\_read\_cam\_registers, 109
  - e\_po3030k\_set\_adc\_offset, 109
  - e\_po3030k\_set\_ae\_speed, 110
  - e\_po3030k\_set\_awb\_ae, 110
  - e\_po3030k\_set\_awb\_ae\_tol, 110
  - e\_po3030k\_set\_bias, 111
  - e\_po3030k\_set\_brigh\_contr, 111
  - e\_po3030k\_set\_cb\_cr\_gain, 111
  - e\_po3030k\_set\_color\_gain, 111
  - e\_po3030k\_set\_color\_matrix, 112
  - e\_po3030k\_set\_color\_mode, 112
  - e\_po3030k\_set\_edge\_prop, 112
  - e\_po3030k\_set\_exposure, 113
  - e\_po3030k\_set\_flicker\_detection, 113
  - e\_po3030k\_set\_flicker\_man\_set, 113
  - e\_po3030k\_set\_flicker\_mode, 114
  - e\_po3030k\_set\_gamma\_coef, 114
  - e\_po3030k\_set\_integr\_time, 115
  - e\_po3030k\_set\_lens\_gain, 115
  - e\_po3030k\_set\_max\_min\_awb, 115
  - e\_po3030k\_set\_max\_min\_exp, 116
  - e\_po3030k\_set\_mirror, 116
  - e\_po3030k\_set\_ref\_exposure, 116
  - e\_po3030k\_set\_register, 116
  - e\_po3030k\_set\_sampling\_mode, 117
  - e\_po3030k\_set\_sepia, 117
  - e\_po3030k\_set\_sepia\_tone, 118
  - e\_po3030k\_set\_speed, 118
  - e\_po3030k\_set\_vsync, 118
  - e\_po3030k\_set\_weight\_win, 119



- e\_po3030k\_set\_ww, [119](#)
- e\_po3030k\_set\_wx, [119](#)
- e\_po3030k\_set\_wy, [120](#)
- e\_po3030k\_sync\_register\_array, [120](#)
- e\_po3030k\_write\_cam\_registers, [122](#)
- e\_po3030k\_write\_gamma\_coef, [122](#)
- MODE\_QQVGA, [107](#)
- MODE\_QVGA, [107](#)
- MODE\_VGA, [107](#)
- PO3030K\_FULL, [107](#)
- SPEED\_128, [107](#)
- SPEED\_16, [107](#)
- SPEED\_2, [107](#)
- SPEED\_2\_3, [107](#)
- SPEED\_32, [107](#)
- SPEED\_4, [107](#)
- SPEED\_64, [107](#)
- SPEED\_8, [107](#)
- fast\_2\_timer/e\_timers.c
  - \_\_attribute\_\_, [186](#)
  - \_poxxxx\_buffer, [187](#)
  - \_poxxxx\_img\_ready, [187](#)
  - blank\_row\_betw, [188](#)
  - e\_poxxxx\_apply\_timer\_config, [186](#)
  - e\_poxxxx\_is\_img\_ready, [187](#)
  - e\_poxxxx\_launch\_capture, [187](#)
  - init\_timer4, [187](#)
  - init\_timer5, [187](#)
- fclose
  - CloseEpuck.m, [241](#)
- fft/e\_fft.c, [225](#)
- fft/e\_fft.h, [226](#)
- fft/e\_fft\_utilities.h, [227](#)
- fft/e\_input\_signal.c, [228](#)
- fft/e\_twiddle\_factors.c, [229](#)
- find\_function
  - e\_agenda\_fast.c, [319](#)
- flow\_led
  - advance\_one\_timer/e\_led.c, [263](#)
  - advance\_one\_timer/e\_led.h, [277](#)
- flush
  - EpuckFlush.m, [242](#)
- flushinput
  - EpuckFlush.m, [242](#)
- flushoutput
  - EpuckFlush.m, [242](#)
- fopen
  - OpenEpuck.m, [245](#)
- friendly\_name
  - BtDevice, [36](#)
- function
  - AgendaType, [35](#)
  - EpuckFlush.m, [242](#)
  - EpuckGetData.m, [243](#)
  - EpuckSendData.m, [244](#)
- fwrite
  - EpuckSendData.m, [244](#)
- GAMMA\_BASE
  - e\_po3030k\_registers.c, [143](#)
  - e\_registers.c, [197](#)
- GAMMASELCOL\_BASE
  - e\_po3030k\_registers.c, [143](#)
  - e\_registers.c, [197](#)
- GRAVITY\_LSM330
  - e\_acc.h, [47](#)
- GRAVITY
  - e\_acc.h, [47](#)
- GREY\_SCALE\_MODE
  - e\_poxxxx.h, [182](#)
  - slow\_3\_timer/e\_po3030k.h, [124](#)
- GROUPDATA\_IN\_PACKET\_OFFSET
  - ComModule.c, [222](#)
- GYRO\_ADDR
  - e\_lsm330.c, [81](#)
- getAllAxesAcc
  - e\_lsm330.c, [81](#)
  - e\_lsm330.h, [83](#)
- getAllAxesAccRaw
  - e\_lsm330.c, [81](#)
  - e\_lsm330.h, [83](#)
- getAllAxesGyro
  - e\_lsm330.c, [81](#)
  - e\_lsm330.h, [83](#)
- getAllAxesGyroRaw
  - e\_lsm330.c, [81](#)
  - e\_lsm330.h, [83](#)
- getBatteryValuePercentage
  - utility.c, [342](#)
  - utility.h, [343](#)
- getBatteryValueRaw
  - utility.c, [342](#)
  - utility.h, [343](#)
- getCameraVersion
  - e\_common.c, [105](#)
- getDiffTimeMs
  - utility.c, [342](#)
  - utility.h, [343](#)
- getDiffTimeMsAndReset
  - utility.c, [342](#)
  - utility.h, [343](#)
- GetHardwareAttenuator
  - ComModule.c, [223](#)
  - ComModule.h, [224](#)
- GetOwnAddress
  - ComModule.c, [223](#)
- GetOwnGroup
  - ComModule.c, [223](#)
  - ComModule.h, [224](#)
- GetRadioEnabledState
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- GetSoftwareAttenuator
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- GetStatus
  - ComModule.c, [223](#)

- ComModule.h, 225
- getTemperature
  - e\_lsm330.c, 81
  - e\_lsm330.h, 83
- getXAxisAcc
  - e\_lsm330.c, 81
  - e\_lsm330.h, 83
- getXAxisGyro
  - e\_lsm330.c, 81
  - e\_lsm330.h, 83
- getYAxisAcc
  - e\_lsm330.c, 81
  - e\_lsm330.h, 83
- getYAxisGyro
  - e\_lsm330.c, 81
  - e\_lsm330.h, 84
- getZAxisAcc
  - e\_lsm330.c, 81
  - e\_lsm330.h, 84
- getZAxisGyro
  - e\_lsm330.c, 81
  - e\_lsm330.h, 84
- getselector
  - utility.c, 342
  - utility.h, 343
- gyroOffsetX
  - e\_lsm330.c, 82
- gyroOffsetY
  - e\_lsm330.c, 82
- gyroOffsetZ
  - e\_lsm330.c, 82
- HARDWAREATT\_SET\_FLAG
  - ComModule.c, 222
- i
  - EpuckGetData.m, 243
- I2C/e\_I2C\_master\_module.c, 229
- I2C/e\_I2C\_master\_module.h, 233
- I2C/e\_I2C\_protocol.c, 237
- I2C/e\_I2C\_protocol.h, 239
- I2C\_ADDR\_CMPS03
  - e\_sensext.h, 218
- I2C\_ADDR\_SENSEXT
  - e\_sensext.h, 218
- I2C\_ADDR\_SRF08
  - e\_sensext.h, 218
- I2C\_ADDR\_SRF10
  - e\_sensext.h, 218
- I2C\_ADDR\_SRF235
  - e\_sensext.h, 218
- I2C, 24
- I\_II
  - e\_remote\_control.h, 314
- IDLE
  - e\_epuck\_ports.h, 329
- INPUT\_PIN
  - e\_epuck\_ports.h, 329
- INTEGR\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- INTERRUPT\_DELAY
  - e\_epuck\_ports.h, 329
- INTERRUPT\_OFF
  - e\_epuck\_ports.h, 329
- INTERRUPT\_ON
  - e\_epuck\_ports.h, 329
- IR0
  - e\_epuck\_ports.h, 329
- IR1
  - e\_epuck\_ports.h, 329
- IR2
  - e\_epuck\_ports.h, 329
- IR3
  - e\_epuck\_ports.h, 329
- IR4
  - e\_epuck\_ports.h, 329
- IR5
  - e\_epuck\_ports.h, 329
- IR6
  - e\_epuck\_ports.h, 329
- IR7
  - e\_epuck\_ports.h, 329
- IRQ\_PIX\_LAT
  - e\_calc\_po3030k.c, 99
  - e\_calc\_po6030k.c, 101
  - e\_calc\_po8030d.c, 102
- idle\_i2c
  - e\_I2C\_master\_module.c, 233
- if
  - EpuckGetData.m, 243
  - EpuckSendData.m, 244
- img\_ready
  - slow\_3\_timer/e\_timers.c, 191
- inclination
  - e\_acc.c, 44
  - TypeAccSpheric, 38
- init\_po8030
  - e\_po8030d.h, 176
  - e\_po8030d\_registers.c, 181
- init\_timer1
  - slow\_3\_timer/e\_timers.c, 190
- init\_timer4
  - fast\_2\_timer/e\_timers.c, 187
  - slow\_3\_timer/e\_timers.c, 190
- init\_timer5
  - fast\_2\_timer/e\_timers.c, 187
  - slow\_3\_timer/e\_timers.c, 190
- init\_tmr1
  - e\_prox.c, 68
- init\_tmr2
  - e\_prox\_timer2.c, 79
- init\_value\_ir
  - advance\_ad\_scan/e\_prox.c, 66
- initAccAndGyro
  - e\_lsm330.c, 81
  - e\_lsm330.h, 84

- InitComModule
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- int16
  - EpuckSendData.m, [244](#)
- is\_ad\_acquisition\_completed
  - advance\_ad\_scan/e\_ad\_conv.c, [52](#)
- is\_ad\_array\_filled
  - advance\_ad\_scan/e\_ad\_conv.c, [52](#)
- isCalibratingFlag
  - e\_acc.c, [44](#)
- isEpuckVersion1\_3
  - e\_init\_port.c, [333](#)
  - e\_init\_port.h, [334](#)
- IsModulePlugged
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- IsPacketReady
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- isPresentFlag
  - e\_init\_port.c, [334](#)
- k2000\_led
  - advance\_one\_timer/e\_led.c, [263](#)
  - advance\_one\_timer/e\_led.h, [278](#)
- LED0
  - e\_epuck\_ports.h, [329](#)
- LED0\_DIR
  - e\_epuck\_ports.h, [329](#)
- LED1
  - e\_epuck\_ports.h, [329](#)
- LED1\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED2
  - e\_epuck\_ports.h, [330](#)
- LED2\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED3
  - e\_epuck\_ports.h, [330](#)
- LED3\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED4
  - e\_epuck\_ports.h, [330](#)
- LED4\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED5
  - e\_epuck\_ports.h, [330](#)
- LED5\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED6
  - e\_epuck\_ports.h, [330](#)
- LED6\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED7
  - e\_epuck\_ports.h, [330](#)
- LED7\_DIR
  - e\_epuck\_ports.h, [330](#)
- LED\_EFFECTS
  - advance\_one\_timer/e\_led.c, [259](#)
- LENSG\_BASE
  - e\_po3030k\_registers.c, [143](#)
  - e\_registers.c, [197](#)
- LIS sensor turret, [21](#)
- LOG2\_BLOCK\_LENGTH
  - e\_fft.h, [227](#)
- lastAccX
  - e\_acc.c, [44](#)
- lastAccY
  - e\_acc.c, [44](#)
- lastAccZ
  - e\_acc.c, [44](#)
- left\_led
  - advance\_one\_timer/e\_led.c, [263](#)
  - advance\_one\_timer/e\_led.h, [278](#)
- left\_motor\_phase
  - advance\_one\_timer/e\_motors.c, [289](#)
  - advance\_one\_timer/fast\_agenda/e\_motors.c, [292](#)
- left\_speed
  - advance\_one\_timer/e\_motors.c, [289](#)
  - advance\_one\_timer/fast\_agenda/e\_motors.c, [292](#)
  - e\_motors.c, [296](#)
  - e\_motors\_timer3.c, [337](#)
- local\_bt\_PIN
  - e\_bluetooth.c, [91](#)
- MAX\_BATT\_VALUE
  - utility.c, [342](#)
- MAXV
  - advance\_one\_timer/e\_motors.c, [286](#)
  - advance\_one\_timer/fast\_agenda/e\_motors.c, [290](#)
- MCLK\_P\_NS
  - e\_po3030k\_registers.c, [143](#)
  - e\_po6030k\_registers.c, [166](#)
  - e\_po8030d\_registers.c, [177](#)
  - e\_registers.c, [197](#)
- MCLK
  - e\_po3030k\_registers.c, [143](#)
  - e\_po6030k\_registers.c, [166](#)
  - e\_po8030d\_registers.c, [177](#)
  - e\_registers.c, [197](#)
- MICO\_BUFFER
  - advance\_ad\_scan/e\_micro.h, [61](#)
- MIC1
  - e\_epuck\_ports.h, [330](#)
- MIC1\_BUFFER
  - advance\_ad\_scan/e\_micro.h, [61](#)
- MIC2
  - e\_epuck\_ports.h, [330](#)
- MIC2\_BUFFER
  - advance\_ad\_scan/e\_micro.h, [61](#)
- MIC3
  - e\_epuck\_ports.h, [330](#)
- MIC\_SAMP\_FREQ
  - advance\_ad\_scan/e\_ad\_conv.h, [55](#)
- MIC\_SAMP\_NB
  - advance\_ad\_scan/e\_ad\_conv.h, [55](#)

- MICRO\_ONLY
  - advance\_ad\_scan/e\_ad\_conv.h, 55
- MICROSEC
  - e\_epuck\_ports.h, 330
- MILLISEC
  - e\_epuck\_ports.h, 330
- MIN\_BATT\_VALUE
  - utility.c, 342
- MINMAXAWB\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- MINMAXEXP\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- MIRROR\_BASE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- MODE\_GRAYSCALE
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- MODE\_QQVGA
  - fast\_2\_timer/e\_po3030k.h, 107
  - slow\_3\_timer/e\_po3030k.h, 124
- MODE\_QVGA
  - fast\_2\_timer/e\_po3030k.h, 107
  - slow\_3\_timer/e\_po3030k.h, 124
- MODE\_R5G6B5
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- MODE\_VGA
  - fast\_2\_timer/e\_po3030k.h, 107
  - slow\_3\_timer/e\_po3030k.h, 124
- MODE\_YUV
  - e\_po3030k\_registers.c, 143
  - e\_registers.c, 197
- MOTOR1\_PHA\_DIR
  - e\_epuck\_ports.h, 330
- MOTOR1\_PHB\_DIR
  - e\_epuck\_ports.h, 330
- MOTOR1\_PHC\_DIR
  - e\_epuck\_ports.h, 331
- MOTOR1\_PHD\_DIR
  - e\_epuck\_ports.h, 331
- MOTOR1\_PHA
  - e\_epuck\_ports.h, 330
- MOTOR1\_PHB
  - e\_epuck\_ports.h, 330
- MOTOR1\_PHC
  - e\_epuck\_ports.h, 330
- MOTOR1\_PHD
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHA\_DIR
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHB\_DIR
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHC\_DIR
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHD\_DIR
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHA
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHB
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHC
  - e\_epuck\_ports.h, 331
- MOTOR2\_PHD
  - e\_epuck\_ports.h, 331
- MUTE
  - e\_remote\_control.h, 314
- Matlab communication, 27
- matlab.c
  - e\_receive\_char\_from\_matlab, 245
  - e\_receive\_int\_from\_matlab, 246
  - e\_send\_char\_to\_matlab, 246
  - e\_send\_int\_to\_matlab, 246
- matlab.h
  - e\_receive\_char\_from\_matlab, 247
  - e\_receive\_int\_from\_matlab, 247
  - e\_send\_char\_to\_matlab, 248
  - e\_send\_int\_to\_matlab, 248
- matlab/matlab files/CloseEpuck.m, 241
- matlab/matlab files/EpuckFlush.m, 241
- matlab/matlab files/EpuckGetData.m, 242
- matlab/matlab files/EpuckSendData.m, 243
- matlab/matlab files/OpenEpuck.m, 244
- matlab/matlab.c, 245
- matlab/matlab.h, 246
- max\_col
  - slow\_3\_timer/e\_timers.c, 191
- max\_row
  - slow\_3\_timer/e\_timers.c, 191
- micro\_only
  - advance\_ad\_scan/e\_ad\_conv.c, 52
- migrate
  - e\_agenda\_fast.c, 319
- motor\_counter\_left
  - e\_motors\_timer3.c, 337
- motor\_counter\_left\_init
  - e\_motors\_timer3.c, 337
- motor\_counter\_right
  - e\_motors\_timer3.c, 337
- motor\_counter\_right\_init
  - e\_motors\_timer3.c, 337
- motor\_led/advance\_one\_timer/e\_agenda.c, 248
- motor\_led/advance\_one\_timer/e\_agenda.h, 252
- motor\_led/advance\_one\_timer/e\_led.c, 258
- motor\_led/advance\_one\_timer/e\_led.h, 271
- motor\_led/advance\_one\_timer/e\_motors.c, 285
- motor\_led/advance\_one\_timer/e\_motors.h, 296
- motor\_led/advance\_one\_timer/e\_remote\_control.c, 310
- motor\_led/advance\_one\_timer/e\_remote\_control.h, 312
- motor\_led/advance\_one\_timer/fast\_agenda/e\_↔
  - agenda\_fast.c, 315
- motor\_led/advance\_one\_timer/fast\_agenda/e\_↔
  - agenda\_fast.h, 319

- motor\_led/advance\_one\_timer/fast\_agenda/e\_led.↔  
c, 264
- motor\_led/advance\_one\_timer/fast\_agenda/e\_led.↔  
h, 278
- motor\_led/advance\_one\_timer/fast\_agenda/e\_↔  
motors.c, 289
- motor\_led/advance\_one\_timer/fast\_agenda/e\_↔  
motors.h, 301
- motor\_led/e\_epuck\_ports.h, 324
- motor\_led/e\_init\_port.c, 332
- motor\_led/e\_init\_port.h, 334
- motor\_led/e\_led.c, 269
- motor\_led/e\_led.h, 283
- motor\_led/e\_motors.c, 293
- motor\_led/e\_motors.h, 305
- motor\_led/e\_motors\_timer3.c, 334
- motors
  - AgendaList, 33
- my\_ceil
  - e\_agenda\_fast.c, 319
- NANOSEC
  - e\_epuck\_ports.h, 331
- NB\_REGISTERS
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- NOP
  - e\_epuck\_ports.h, 331
- nbr\_pas\_left
  - e\_motors.c, 296
  - e\_motors\_timer3.c, 337
- nbr\_pas\_right
  - e\_motors.c, 296
  - e\_motors\_timer3.c, 337
- nbr\_steps\_left
  - advance\_one\_timer/e\_motors.c, 289
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
- nbr\_steps\_right
  - advance\_one\_timer/e\_motors.c, 289
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
- next
  - AgendaType, 35
- number
  - BtEPuck, 36
- OPERATION\_OK
  - e\_i2C\_master\_module.h, 234
- OUT\_AUX\_1
  - e\_remote\_control.h, 314
- OUTPUT\_PIN
  - e\_epuck\_ports.h, 331
- OWNADDRH\_REG\_ADDR
  - ComModule.c, 222
- OWNADDRL\_REG\_ADDR
  - ComModule.c, 222
- OWNGROUP\_REG\_ADDR
  - ComModule.c, 222
- OpenEpuck.m
  - EpuckPort, 245
- Error, 245
- fopen, 245
- port, 245
- return, 245
- orientation
  - e\_acc.c, 44
  - TypeAccSpheric, 38
- PACKET\_LOST\_FLAG
  - ComModule.c, 222
- PACKET\_READY\_FLAG
  - ComModule.c, 222
- PLL
  - e\_epuck\_ports.h, 331
- PO3030K\_FULL
  - fast\_2\_timer/e\_po3030k.h, 107
  - slow\_3\_timer/e\_po3030k.h, 124
- PO\_6030\_MODE\_QQVGA
  - e\_po6030k.h, 159
- PO\_6030\_MODE\_QVGA
  - e\_po6030k.h, 159
- PO\_6030\_MODE\_VGA
  - e\_po6030k.h, 159
- PO\_6030\_SPEED\_1
  - e\_po6030k.h, 159
- PO\_6030\_SPEED\_2
  - e\_po6030k.h, 159
- PO\_6030\_SPEED\_4
  - e\_po6030k.h, 159
- PO\_6030\_SPEED\_8
  - e\_po6030k.h, 159
- PO\_8030\_BAYER\_CLOCK\_1
  - e\_po8030d.h, 171
- PO\_8030\_BAYER\_CLOCK\_2
  - e\_po8030d.h, 171
- PO\_8030\_BAYER\_CLOCK\_4
  - e\_po8030d.h, 171
- PO\_8030\_BAYER\_CLOCK\_8
  - e\_po8030d.h, 171
- PO\_8030\_MODE\_QQVGA
  - e\_po8030d.h, 171
- PO\_8030\_MODE\_QVGA
  - e\_po8030d.h, 171
- PO\_8030\_MODE\_VGA
  - e\_po8030d.h, 171
- PO\_8030\_SPEED\_1
  - e\_po8030d.h, 171
- PO\_8030\_SPEED\_2
  - e\_po8030d.h, 171
- PO\_8030\_SPEED\_4
  - e\_po8030d.h, 171
- PO\_8030\_SPEED\_8
  - e\_po8030d.h, 171
- POWERSAVE
  - advance\_one\_timer/e\_motors.c, 286
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 290
- POXXXX\_FULL
  - e\_poxxxx.h, 182
- PULSE\_IR0

- e\_epuck\_ports.h, 331
- PULSE\_IR0\_DIR
  - e\_epuck\_ports.h, 331
- PULSE\_IR1
  - e\_epuck\_ports.h, 331
- PULSE\_IR1\_DIR
  - e\_epuck\_ports.h, 331
- PULSE\_IR2
  - e\_epuck\_ports.h, 331
- PULSE\_IR2\_DIR
  - e\_epuck\_ports.h, 331
- PULSE\_IR3
  - e\_epuck\_ports.h, 331
- PULSE\_IR3\_DIR
  - e\_epuck\_ports.h, 331
- PULSE\_LENIGHT
  - advance\_ad\_scan/e\_ad\_conv.h, 55
- PULSE\_PERIOD
  - advance\_ad\_scan/e\_ad\_conv.h, 55
- pbp\_current
  - slow\_3\_timer/e\_timers.c, 191
- pixel\_betw\_pixel
  - slow\_3\_timer/e\_timers.c, 191
- po3030k\_get\_pixelclock
  - e\_po3030k\_registers.c, 157
  - e\_registers.c, 210
- port
  - OpenEpuck.m, 245
- Ports, motors and LEDs, 28
- RADIO\_ENABLED\_FLAG
  - ComModule.c, 222
- READ
  - e\_I2C\_master\_module.h, 234
- REC\_BUFFER\_END
  - ComModule.c, 222
- REC\_BUFFER\_START
  - ComModule.c, 222
- REFREPO\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- REMOTE\_DIR
  - e\_epuck\_ports.h, 332
- REMOTE
  - e\_epuck\_ports.h, 332
- REQUEST\_TO\_SEND\_FLAG
  - ComModule.c, 222
- RESET
  - e\_epuck\_ports.h, 332
- RESTART
  - e\_I2C\_master\_module.h, 234
- RGB\_565\_MODE
  - e\_poxxxx.h, 182
  - slow\_3\_timer/e\_po3030k.h, 124
- Radio communication, 22
- rawToDpms
  - e\_lsm330.c, 81
  - e\_lsm330.h, 84
- rawToDps
  - e\_lsm330.c, 81
  - e\_lsm330.h, 84
- readReg
  - e\_lsm330.c, 81
- readRegMulti
  - e\_lsm330.c, 82
- ReadRegister
  - ComModule.c, 223
- readee
  - e\_common.c, 105
- receivedFormat
  - EpuckGetData.m, 243
- recompute\_speeds
  - e\_agenda\_fast.c, 319
- reflected\_ir
  - e\_prox.c, 69
  - e\_prox\_timer2.c, 79
- resetTime
  - utility.c, 342
  - utility.h, 343
- return
  - EpuckGetData.m, 243
  - EpuckSendData.m, 244
  - OpenEpuck.m, 245
- right\_led
  - advance\_one\_timer/e\_led.c, 264
  - advance\_one\_timer/e\_led.h, 278
- right\_motor\_phase
  - advance\_one\_timer/e\_motors.c, 289
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
- right\_speed
  - advance\_one\_timer/e\_motors.c, 289
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 293
  - e\_motors.c, 296
  - e\_motors\_timer3.c, 338
- run\_left\_motor
  - advance\_one\_timer/e\_motors.c, 288
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
- run\_right\_motor
  - advance\_one\_timer/e\_motors.c, 288
  - advance\_one\_timer/fast\_agenda/e\_motors.c, 292
- SAMPLING\_ADDR
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- SELECTOR0
  - e\_epuck\_ports.h, 332
- SELECTOR0\_DIR
  - e\_epuck\_ports.h, 332
- SELECTOR1
  - e\_epuck\_ports.h, 332
- SELECTOR1\_DIR
  - e\_epuck\_ports.h, 332
- SELECTOR2
  - e\_epuck\_ports.h, 332
- SELECTOR2\_DIR
  - e\_epuck\_ports.h, 332
- SELECTOR3
  - e\_epuck\_ports.h, 332

- SELECTOR3\_DIR
  - e\_epuck\_ports.h, [332](#)
- SEND\_BUFFER\_END
  - ComModule.c, [222](#)
- SEND\_BUFFER\_START
  - ComModule.c, [222](#)
- SEND\_REG\_ADDR
  - ComModule.c, [222](#)
- SEPIA\_BASE
  - e\_po3030k\_registers.c, [144](#)
  - e\_registers.c, [197](#)
- SEPIATONE\_BASE
  - e\_po3030k\_registers.c, [144](#)
  - e\_registers.c, [197](#)
- SHARP\_LED1
  - e\_sharp.h, [220](#)
- SHARP\_LED1\_DIR
  - e\_sharp.h, [220](#)
- SHARP\_LED2
  - e\_sharp.h, [220](#)
- SHARP\_LED2\_DIR
  - e\_sharp.h, [220](#)
- SHARP\_LED3
  - e\_sharp.h, [220](#)
- SHARP\_LED3\_DIR
  - e\_sharp.h, [220](#)
- SHARP\_LED4
  - e\_sharp.h, [220](#)
- SHARP\_LED4\_DIR
  - e\_sharp.h, [220](#)
- SHARP\_LED5
  - e\_sharp.h, [220](#)
- SHARP\_LED5\_DIR
  - e\_sharp.h, [220](#)
- SHARP\_VIN\_DIR
  - e\_sharp.h, [220](#)
- SHARP\_VIN
  - e\_sharp.h, [220](#)
- SHARP
  - e\_sharp.h, [220](#)
- SIO\_C\_DIR
  - e\_epuck\_ports.h, [332](#)
- SIO\_D\_DIR
  - e\_epuck\_ports.h, [332](#)
- SIO\_C
  - e\_epuck\_ports.h, [332](#)
- SIO\_D
  - e\_epuck\_ports.h, [332](#)
- SLEEP
  - e\_epuck\_ports.h, [332](#)
- SOFTATT\_REG\_ADDR
  - ComModule.c, [222](#)
- SPEED\_128
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_16
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_2
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_2\_3
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_32
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_4
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_64
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_8
  - fast\_2\_timer/e\_po3030k.h, [107](#)
  - slow\_3\_timer/e\_po3030k.h, [125](#)
- SPEED\_ADDR
  - e\_po3030k\_registers.c, [144](#)
  - e\_registers.c, [197](#)
- STANDBY
  - e\_remote\_control.h, [314](#)
- START
  - e\_I2C\_master\_module.h, [234](#)
- STATUS\_REG\_ADDR
  - ComModule.c, [223](#)
- STOP\_TMR1
  - e\_epuck\_ports.h, [332](#)
- STOP\_TMR2
  - e\_epuck\_ports.h, [332](#)
- STOP\_TMR3
  - e\_epuck\_ports.h, [332](#)
- STOP\_TMR4
  - e\_epuck\_ports.h, [332](#)
- STOP\_TMR5
  - e\_epuck\_ports.h, [332](#)
- STOP
  - e\_I2C\_master\_module.h, [234](#)
- search\_best\_fit
  - e\_agenda\_fast.c, [319](#)
- selector
  - advance\_ad\_scan/e\_ad\_conv.c, [52](#)
- SendPacket
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- sensex\_t\_wait
  - e\_sensex.c, [217](#)
- SetHardwareAttenuator
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- SetOwnAddress
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- SetOwnGroup
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- SetRadioEnabledState

- ComModule.c, [223](#)
- ComModule.h, [225](#)
- SetSoftwareAttenuator
  - ComModule.c, [223](#)
  - ComModule.h, [225](#)
- size
  - EpuckSendData.m, [244](#)
- slow\_3\_timer/e\_po3030k.h
  - ARRAY\_HEIGHT, [124](#)
  - ARRAY\_WIDTH, [124](#)
  - e\_po3030k\_apply\_timer\_config, [125](#)
  - e\_po3030k\_config\_cam, [125](#)
  - e\_po3030k\_get\_bytes\_per\_pixel, [126](#)
  - e\_po3030k\_get\_register, [127](#)
  - e\_po3030k\_init\_cam, [127](#)
  - e\_po3030k\_is\_img\_ready, [127](#)
  - e\_po3030k\_launch\_capture, [127](#)
  - e\_po3030k\_read\_cam\_registers, [128](#)
  - e\_po3030k\_set\_adc\_offset, [128](#)
  - e\_po3030k\_set\_ae\_speed, [128](#)
  - e\_po3030k\_set\_awb\_ae, [128](#)
  - e\_po3030k\_set\_awb\_ae\_tol, [129](#)
  - e\_po3030k\_set\_bias, [129](#)
  - e\_po3030k\_set\_brigh\_contr, [129](#)
  - e\_po3030k\_set\_cb\_cr\_gain, [130](#)
  - e\_po3030k\_set\_color\_gain, [130](#)
  - e\_po3030k\_set\_color\_matrix, [130](#)
  - e\_po3030k\_set\_color\_mode, [130](#)
  - e\_po3030k\_set\_edge\_prop, [131](#)
  - e\_po3030k\_set\_exposure, [131](#)
  - e\_po3030k\_set\_flicker\_detection, [131](#)
  - e\_po3030k\_set\_flicker\_man\_set, [132](#)
  - e\_po3030k\_set\_flicker\_mode, [132](#)
  - e\_po3030k\_set\_gamma\_coef, [133](#)
  - e\_po3030k\_set\_integr\_time, [133](#)
  - e\_po3030k\_set\_lens\_gain, [133](#)
  - e\_po3030k\_set\_max\_min\_awb, [134](#)
  - e\_po3030k\_set\_max\_min\_exp, [134](#)
  - e\_po3030k\_set\_mirror, [134](#)
  - e\_po3030k\_set\_ref\_exposure, [135](#)
  - e\_po3030k\_set\_register, [135](#)
  - e\_po3030k\_set\_sampling\_mode, [135](#)
  - e\_po3030k\_set\_sepia, [136](#)
  - e\_po3030k\_set\_sepia\_tone, [136](#)
  - e\_po3030k\_set\_speed, [136](#)
  - e\_po3030k\_set\_vsync, [137](#)
  - e\_po3030k\_set\_weight\_win, [137](#)
  - e\_po3030k\_set\_ww, [138](#)
  - e\_po3030k\_set\_wx, [138](#)
  - e\_po3030k\_set\_wy, [138](#)
  - e\_po3030k\_sync\_register\_array, [139](#)
  - e\_po3030k\_write\_cam\_registers, [139](#)
  - e\_po3030k\_write\_gamma\_coef, [139](#)
  - GREY\_SCALE\_MODE, [124](#)
  - MODE\_QQVGA, [124](#)
  - MODE\_QVGA, [124](#)
  - MODE\_VGA, [124](#)
  - PO3030K\_FULL, [124](#)
  - RGB\_565\_MODE, [124](#)
  - SPEED\_128, [125](#)
  - SPEED\_16, [125](#)
  - SPEED\_2, [125](#)
  - SPEED\_2\_3, [125](#)
  - SPEED\_32, [125](#)
  - SPEED\_4, [125](#)
  - SPEED\_64, [125](#)
  - SPEED\_8, [125](#)
  - YUV\_MODE, [125](#)
  - slow\_3\_timer/e\_timers.c
    - \_\_attribute\_\_, [189](#)
    - bbl\_current, [190](#)
    - blank\_betw\_lines, [190](#)
    - bpp\_current, [190](#)
    - buf\_pos, [190](#)
    - buffer, [190](#)
    - bytes\_per\_pixel, [190](#)
    - current\_col, [190](#)
    - current\_row, [191](#)
    - e\_po3030k\_apply\_timer\_config, [189](#)
    - e\_po3030k\_is\_img\_ready, [189](#)
    - e\_po3030k\_launch\_capture, [189](#)
    - img\_ready, [191](#)
    - init\_timer1, [190](#)
    - init\_timer4, [190](#)
    - init\_timer5, [190](#)
    - max\_col, [191](#)
    - max\_row, [191](#)
    - pbp\_current, [191](#)
    - pixel\_betw\_pixel, [191](#)
  - snake\_led
    - advance\_one\_timer/e\_led.c, [264](#)
    - advance\_one\_timer/e\_led.h, [278](#)
  - Sound, [19](#)
  - speed
    - AgendaList, [33](#)
  - TCY\_PIC
    - e\_epuck\_ports.h, [332](#)
  - TRESHV
    - advance\_one\_timer/e\_motors.c, [286](#)
    - advance\_one\_timer/fast\_agenda/e\_motors.c, [290](#)
  - TRUE
    - e\_epuck\_ports.h, [332](#)
  - TX\_IDLE\_FLAG
    - ComModule.c, [223](#)
  - TX\_SEND\_ERROR
    - ComModule.c, [223](#)
  - TYPEDATA\_IN\_PACKET\_OFFSET
    - ComModule.c, [223](#)
  - testAccGyroPresence
    - e\_init\_port.c, [334](#)
  - testWriteReg
    - e\_po8030d.h, [176](#)
    - e\_po8030d\_registers.c, [181](#)
  - tickAdclsr
    - advance\_ad\_scan/e\_ad\_conv.c, [52](#)
    - utility.c, [343](#)



- timers\_in\_use
  - AgendaList, 33
- TypeAccRaw, 37
  - acc\_x, 37
  - acc\_y, 37
  - acc\_z, 37
- TypeAccSpheric, 37
  - acceleration, 38
  - inclination, 38
  - orientation, 38
- UART, 31
- uart/e\_epuck\_ports.inc, 338
- uart/e\_uart\_char.h, 338
- updateAccl2CCounter
  - advance\_ad\_scan/e\_ad\_conv.c, 52
  - e\_acc.c, 44
- utility.c
  - BATT\_VALUES\_RANGE, 342
  - e\_acc\_scan, 342
  - e\_last\_acc\_scan\_id, 342
  - getBatteryValuePercentage, 342
  - getBatteryValueRaw, 342
  - getDiffTimeMs, 342
  - getDiffTimeMsAndReset, 342
  - getselector, 342
  - MAX\_BATT\_VALUE, 342
  - MIN\_BATT\_VALUE, 342
  - resetTime, 342
  - tickAdclsr, 343
  - wait, 342
- utility.h
  - getBatteryValuePercentage, 343
  - getBatteryValueRaw, 343
  - getDiffTimeMs, 343
  - getDiffTimeMsAndReset, 343
  - getselector, 343
  - resetTime, 343
  - wait, 343
- utility/utility.c, 341
- utility/utility.h, 343
- VOL\_DOWN
  - e\_remote\_control.h, 314
- VOL\_UP
  - e\_remote\_control.h, 314
- VSYNCCOL\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- VSYNCSTART\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- VSYNCTOP\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- WEIGHWIN\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- WINDOW\_X1\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 197
- WINDOW\_X2\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 198
- WINDOW\_Y1\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 198
- WINDOW\_Y2\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 198
- WRITE
  - e\_I2C\_master\_module.h, 234
- WW\_BASE
  - e\_po3030k\_registers.c, 144
  - e\_registers.c, 198
- wait
  - utility.c, 342
  - utility.h, 343
- waiting
  - AgendaList, 34
- while
  - EpuckGetData.m, 243
- writeReg
  - e\_lsm330.c, 82
- WriteRegister
  - ComModule.c, 223
- YUV\_MODE
  - e\_poxxxx.h, 182
  - slow\_3\_timer/e\_po3030k.h, 125