

Trabajo_1_Micros

Generado por Doxygen 1.8.10

Sábado, 19 de Septiembre de 2015 04:57:53

Índice general

1	Trabajo_1_Micros	1
2	Índice de estructura de datos	3
2.1	Estructura de datos	3
3	Índice de archivos	5
3.1	Lista de archivos	5
4	Documentación de las estructuras de datos	7
4.1	Referencia de la Estructura _win	7
4.1.1	Documentación de los campos	7
4.1.1.1	_attrs	7
4.1.1.2	_begx	7
4.1.1.3	_begy	8
4.1.1.4	_bkgd	8
4.1.1.5	_bmarg	8
4.1.1.6	_clear	8
4.1.1.7	_curx	8
4.1.1.8	_cury	8
4.1.1.9	_delayms	8
4.1.1.10	_firstch	8
4.1.1.11	_flags	8
4.1.1.12	_immed	8
4.1.1.13	_lastch	8
4.1.1.14	_leaveit	8
4.1.1.15	_maxx	8
4.1.1.16	_maxy	8
4.1.1.17	_nodelay	8
4.1.1.18	_parent	8
4.1.1.19	_parx	8
4.1.1.20	_pary	8
4.1.1.21	_scroll	8

4.1.1.22	_sync	8
4.1.1.23	_tmarg	8
4.1.1.24	_use_keypad	8
4.1.1.25	_y	8
4.2	Referencia de la Estructura ins_t	8
4.2.1	Documentación de los campos	9
4.2.1.1	array	9
4.3	Referencia de la Estructura instruction_t	9
4.3.1	Documentación de los campos	9
4.3.1.1	mnemonic	9
4.3.1.2	op1_type	9
4.3.1.3	op1_value	9
4.3.1.4	op2_type	9
4.3.1.5	op2_value	9
4.3.1.6	op3_type	9
4.3.1.7	op3_value	9
4.4	Referencia de la Estructura MEVENT	9
4.4.1	Documentación de los campos	10
4.4.1.1	bstate	10
4.4.1.2	id	10
4.4.1.3	x	10
4.4.1.4	y	10
4.4.1.5	z	10
4.5	Referencia de la Estructura MOUSE_STATUS	10
4.5.1	Documentación de los campos	10
4.5.1.1	button	10
4.5.1.2	changes	10
4.5.1.3	x	10
4.5.1.4	y	10
4.6	Referencia de la Estructura SCREEN	10
4.6.1	Documentación de los campos	11
4.6.1.1	_map_mbe_to_key	11
4.6.1.2	_preserve	11
4.6.1.3	_restore	11
4.6.1.4	_trap_mbe	11
4.6.1.5	alive	11
4.6.1.6	audible	11
4.6.1.7	autocr	11
4.6.1.8	cbreak	11
4.6.1.9	cols	11

4.6.1.10	<code>curscol</code>	12
4.6.1.11	<code>cursrow</code>	12
4.6.1.12	<code>delaytenths</code>	12
4.6.1.13	<code>echo</code>	12
4.6.1.14	<code>key_code</code>	12
4.6.1.15	<code>line_color</code>	12
4.6.1.16	<code>lines</code>	12
4.6.1.17	<code>linesrippedoff</code>	12
4.6.1.18	<code>linesrippedoffontop</code>	12
4.6.1.19	<code>mono</code>	12
4.6.1.20	<code>mouse_wait</code>	12
4.6.1.21	<code>orig_attr</code>	12
4.6.1.22	<code>orig_back</code>	12
4.6.1.23	<code>orig_cursor</code>	12
4.6.1.24	<code>orig_fore</code>	12
4.6.1.25	<code>raw_inp</code>	12
4.6.1.26	<code>raw_out</code>	12
4.6.1.27	<code>resized</code>	12
4.6.1.28	<code>return_key_modifiers</code>	12
4.6.1.29	<code>save_key_modifiers</code>	12
4.6.1.30	<code>slk_winpnr</code>	12
4.6.1.31	<code>slklines</code>	12
4.6.1.32	<code>visibility</code>	12
5	Documentación de archivos	13
5.1	Referencia del Archivo ALU.c	13
5.1.1	Documentación de los 'defines'	14
5.1.1.1	<code>C</code>	14
5.1.1.2	<code>N</code>	14
5.1.1.3	<code>PC</code>	14
5.1.1.4	<code>V</code>	14
5.1.1.5	<code>Z</code>	14
5.1.2	Documentación de las funciones	14
5.1.2.1	<code>ADC(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)</code>	14
5.1.2.2	<code>ADD(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)</code>	14
5.1.2.3	<code>AND(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)</code>	15
5.1.2.4	<code>Bandera_C(uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)</code>	15
5.1.2.5	<code>Bandera_N(uint32_t Rd, char *R_Banderas)</code>	15

5.1.2.6	Bandera_V(uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	15
5.1.2.7	Bandera_Z(uint32_t Rd, char *R_Banderas)	15
5.1.2.8	Banderas(uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	16
5.1.2.9	CMN(uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)	16
5.1.2.10	CMP(uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)	16
5.1.2.11	EOR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	16
5.1.2.12	MOV(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)	16
5.1.2.13	MUL(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	17
5.1.2.14	ORR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	17
5.1.2.15	SBC(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	17
5.1.2.16	SUB(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	17
5.1.2.17	TST(uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)	18
5.2	Referencia del Archivo ALU.h	18
5.2.1	Documentación de las funciones	19
5.2.1.1	ADC(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	19
5.2.1.2	ADD(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	19
5.2.1.3	AND(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	19
5.2.1.4	Bandera_C(uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	19
5.2.1.5	Bandera_N(uint32_t Rd, char *R_Banderas)	19
5.2.1.6	Bandera_V(uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	20
5.2.1.7	Bandera_Z(uint32_t Rd, char *R_Banderas)	20
5.2.1.8	Banderas(uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	20
5.2.1.9	CMN(uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)	20
5.2.1.10	CMP(uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)	20
5.2.1.11	EOR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	21
5.2.1.12	MOV(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)	21
5.2.1.13	MUL(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	21
5.2.1.14	ORR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	21
5.2.1.15	SBC(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	22
5.2.1.16	SUB(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↵ Banderas)	22
5.2.1.17	TST(uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)	22

5.3	Referencia del Archivo branch.c	22
5.3.1	Documentación de los 'defines'	23
5.3.1.1	C	23
5.3.1.2	LR	23
5.3.1.3	N	23
5.3.1.4	PC	23
5.3.1.5	V	23
5.3.1.6	Z	24
5.3.2	Documentación de las funciones	24
5.3.2.1	B(uint32_t *Registro, uint32_t Label)	24
5.3.2.2	BAL(uint32_t *Registro, uint32_t Label)	24
5.3.2.3	BCC(char *R_Banderas, uint32_t *Registro, uint32_t Label)	24
5.3.2.4	BCS(char *R_Banderas, uint32_t *Registro, uint32_t Label)	24
5.3.2.5	BEQ(char *R_Banderas, uint32_t *Registro, uint32_t Label)	24
5.3.2.6	BGE(char *R_Banderas, uint32_t *Registro, uint32_t Label)	25
5.3.2.7	BGT(char *R_Banderas, uint32_t *Registro, uint32_t Label)	25
5.3.2.8	BHI(char *R_Banderas, uint32_t *Registro, uint32_t Label)	25
5.3.2.9	BL(uint32_t *Registro, uint32_t Label)	25
5.3.2.10	BLE(char *R_Banderas, uint32_t *Registro, uint32_t Label)	25
5.3.2.11	BLS(char *R_Banderas, uint32_t *Registro, uint32_t Label)	25
5.3.2.12	BLT(char *R_Banderas, uint32_t *Registro, uint32_t Label)	26
5.3.2.13	BMI(char *R_Banderas, uint32_t *Registro, uint32_t Label)	26
5.3.2.14	BNE(char *R_Banderas, uint32_t *Registro, uint32_t Label)	26
5.3.2.15	BPL(char *R_Banderas, uint32_t *Registro, uint32_t Label)	26
5.3.2.16	BVC(char *R_Banderas, uint32_t *Registro, uint32_t Label)	26
5.3.2.17	BVS(char *R_Banderas, uint32_t *Registro, uint32_t Label)	27
5.3.2.18	BX(uint32_t *Registro)	27
5.4	Referencia del Archivo branch.h	27
5.4.1	Documentación de las funciones	28
5.4.1.1	B(uint32_t *Registro, uint32_t Label)	28
5.4.1.2	BAL(uint32_t *Registro, uint32_t Label)	28
5.4.1.3	BCC(char *R_Banderas, uint32_t *Registro, uint32_t Label)	28
5.4.1.4	BCS(char *R_Banderas, uint32_t *Registro, uint32_t Label)	28
5.4.1.5	BEQ(char *R_Banderas, uint32_t *Registro, uint32_t Label)	29
5.4.1.6	BGE(char *R_Banderas, uint32_t *Registro, uint32_t Label)	29
5.4.1.7	BGT(char *R_Banderas, uint32_t *Registro, uint32_t Label)	29
5.4.1.8	BHI(char *R_Banderas, uint32_t *Registro, uint32_t Label)	29
5.4.1.9	BL(uint32_t *Registro, uint32_t Label)	29
5.4.1.10	BLE(char *R_Banderas, uint32_t *Registro, uint32_t Label)	30
5.4.1.11	BLS(char *R_Banderas, uint32_t *Registro, uint32_t Label)	30

5.4.1.12	BLT(char *R_Banderas, uint32_t *Registro, uint32_t Label)	30
5.4.1.13	BMI(char *R_Banderas, uint32_t *Registro, uint32_t Label)	30
5.4.1.14	BNE(char *R_Banderas, uint32_t *Registro, uint32_t Label)	30
5.4.1.15	BPL(char *R_Banderas, uint32_t *Registro, uint32_t Label)	31
5.4.1.16	BVC(char *R_Banderas, uint32_t *Registro, uint32_t Label)	31
5.4.1.17	BVS(char *R_Banderas, uint32_t *Registro, uint32_t Label)	31
5.4.1.18	BX(uint32_t *Registro)	31
5.5	Referencia del Archivo colors.h	31
5.5.1	Documentación de los 'defines'	32
5.5.1.1	AQUA	32
5.5.1.2	BLACK	32
5.5.1.3	BLUE	32
5.5.1.4	BRIGHT_WHITE	32
5.5.1.5	GRAY	32
5.5.1.6	GREEN	32
5.5.1.7	LIGHT_AQUA	32
5.5.1.8	LIGHT_BLUE	32
5.5.1.9	LIGHT_GREEN	32
5.5.1.10	LIGHT_PURPLE	32
5.5.1.11	LIGHT_RED	32
5.5.1.12	LIGHT_YELLOW	32
5.5.1.13	PURPLE	32
5.5.1.14	RED	32
5.5.1.15	WHITE	32
5.5.1.16	YELLOW	32
5.6	Referencia del Archivo curses.h	32
5.6.1	Documentación de los 'defines'	47
5.6.1.1	A_ALTCHARSET	47
5.6.1.2	A_ATTRIBUTES	47
5.6.1.3	A_BLINK	47
5.6.1.4	A_BOLD	47
5.6.1.5	A_BUTTON_CHANGED	47
5.6.1.6	A_CHARTEXT	47
5.6.1.7	A_COLOR	47
5.6.1.8	A_DIM	47
5.6.1.9	A_INVIS	47
5.6.1.10	A_ITALIC	47
5.6.1.11	A_LEFTLINE	47
5.6.1.12	A_NORMAL	47
5.6.1.13	A_PROTECT	47

5.6.1.14	A_REVERSE	47
5.6.1.15	A_RIGHTLINE	47
5.6.1.16	A_STANDOUT	47
5.6.1.17	A_UNDERLINE	47
5.6.1.18	ACS_BBSS	47
5.6.1.19	ACS_BLOCK	47
5.6.1.20	ACS_BOARD	47
5.6.1.21	ACS_BSBS	48
5.6.1.22	ACS_BSSB	48
5.6.1.23	ACS_BSSS	48
5.6.1.24	ACS_BTEE	48
5.6.1.25	ACS_BULLET	48
5.6.1.26	ACS_CKBOARD	48
5.6.1.27	ACS_DARROW	48
5.6.1.28	ACS_DEGREE	48
5.6.1.29	ACS_DIAMOND	48
5.6.1.30	ACS_GEQUAL	48
5.6.1.31	ACS_HLINE	48
5.6.1.32	ACS_LANTERN	48
5.6.1.33	ACS_LARROW	48
5.6.1.34	ACS_LEQUAL	48
5.6.1.35	ACS_LLCORNER	48
5.6.1.36	ACS_LRCORNER	48
5.6.1.37	ACS_LTEE	48
5.6.1.38	ACS_NEQUAL	48
5.6.1.39	ACS_PI	48
5.6.1.40	ACS_PICK	48
5.6.1.41	ACS_PLMINUS	48
5.6.1.42	ACS_PLUS	48
5.6.1.43	ACS_RARROW	48
5.6.1.44	ACS_RTEE	48
5.6.1.45	ACS_S1	48
5.6.1.46	ACS_S3	48
5.6.1.47	ACS_S7	48
5.6.1.48	ACS_S9	48
5.6.1.49	ACS_SBBS	49
5.6.1.50	ACS_SBSB	49
5.6.1.51	ACS_SBSS	49
5.6.1.52	ACS_SSBB	49
5.6.1.53	ACS_SSBS	49

5.6.1.54	ACS_SSSB	49
5.6.1.55	ACS_SSSS	49
5.6.1.56	ACS_STERLING	49
5.6.1.57	ACS_TTEE	49
5.6.1.58	ACS_UARROW	49
5.6.1.59	ACS_ULCORNER	49
5.6.1.60	ACS_URCORNER	49
5.6.1.61	ACS_VLINE	49
5.6.1.62	ALL_MOUSE_EVENTS	49
5.6.1.63	ALT_0	49
5.6.1.64	ALT_1	49
5.6.1.65	ALT_2	49
5.6.1.66	ALT_3	49
5.6.1.67	ALT_4	49
5.6.1.68	ALT_5	49
5.6.1.69	ALT_6	49
5.6.1.70	ALT_7	49
5.6.1.71	ALT_8	49
5.6.1.72	ALT_9	49
5.6.1.73	ALT_A	49
5.6.1.74	ALT_B	49
5.6.1.75	ALT_BKSP	49
5.6.1.76	ALT_BQUOTE	49
5.6.1.77	ALT_BSLASH	50
5.6.1.78	ALT_C	50
5.6.1.79	ALT_COMMA	50
5.6.1.80	ALT_D	50
5.6.1.81	ALT_DEL	50
5.6.1.82	ALT_DOWN	50
5.6.1.83	ALT_E	50
5.6.1.84	ALT_END	50
5.6.1.85	ALT_ENTER	50
5.6.1.86	ALT_EQUAL	50
5.6.1.87	ALT_ESC	50
5.6.1.88	ALT_F	50
5.6.1.89	ALT_FQUOTE	50
5.6.1.90	ALT_FSLASH	50
5.6.1.91	ALT_G	50
5.6.1.92	ALT_H	50
5.6.1.93	ALT_HOME	50

5.6.1.94 ALT_I	50
5.6.1.95 ALT_INS	50
5.6.1.96 ALT_J	50
5.6.1.97 ALT_K	50
5.6.1.98 ALT_L	50
5.6.1.99 ALT_LBRACKET	50
5.6.1.100 ALT_LEFT	50
5.6.1.101 ALT_M	50
5.6.1.102 ALT_MINUS	50
5.6.1.103 ALT_N	50
5.6.1.104 ALT_O	50
5.6.1.105 ALT_P	51
5.6.1.106 ALT_PAD0	51
5.6.1.107 ALT_PAD1	51
5.6.1.108 ALT_PAD2	51
5.6.1.109 ALT_PAD3	51
5.6.1.110 ALT_PAD4	51
5.6.1.111 ALT_PAD5	51
5.6.1.112 ALT_PAD6	51
5.6.1.113 ALT_PAD7	51
5.6.1.114 ALT_PAD8	51
5.6.1.115 ALT_PAD9	51
5.6.1.116 ALT_PADENTER	51
5.6.1.117 ALT_PADMINUS	51
5.6.1.118 ALT_PADPLUS	51
5.6.1.119 ALT_PADSLASH	51
5.6.1.120 ALT_PADSTAR	51
5.6.1.121 ALT_PADSTOP	51
5.6.1.122 ALT_PGDN	51
5.6.1.123 ALT_PGUP	51
5.6.1.124 ALT_Q	51
5.6.1.125 ALT_R	51
5.6.1.126 ALT_RBRACKET	51
5.6.1.127 ALT_RIGHT	51
5.6.1.128 ALT_S	51
5.6.1.129 ALT_SEMICOLON	51
5.6.1.130 ALT_STOP	51
5.6.1.131 ALT_T	51
5.6.1.132 ALT_TAB	51
5.6.1.133 ALT_U	52

5.6.1.134 ALT_UP	52
5.6.1.135 ALT_V	52
5.6.1.136 ALT_W	52
5.6.1.137 ALT_X	52
5.6.1.138 ALT_Y	52
5.6.1.139 ALT_Z	52
5.6.1.140 ATR_MSK	52
5.6.1.141 ATR_NRM	52
5.6.1.142 BSDcurses	52
5.6.1.143 BUTTON1_CLICKED	52
5.6.1.144 BUTTON1_DOUBLE_CLICKED	52
5.6.1.145 BUTTON1_MOVED	52
5.6.1.146 BUTTON1_PRESSED	52
5.6.1.147 BUTTON1_RELEASED	52
5.6.1.148 BUTTON1_TRIPLE_CLICKED	52
5.6.1.149 BUTTON2_CLICKED	52
5.6.1.150 BUTTON2_DOUBLE_CLICKED	52
5.6.1.151 BUTTON2_MOVED	52
5.6.1.152 BUTTON2_PRESSED	52
5.6.1.153 BUTTON2_RELEASED	52
5.6.1.154 BUTTON2_TRIPLE_CLICKED	52
5.6.1.155 BUTTON3_CLICKED	52
5.6.1.156 BUTTON3_DOUBLE_CLICKED	52
5.6.1.157 BUTTON3_MOVED	52
5.6.1.158 BUTTON3_PRESSED	52
5.6.1.159 BUTTON3_RELEASED	52
5.6.1.160 BUTTON3_TRIPLE_CLICKED	52
5.6.1.161 BUTTON4_CLICKED	53
5.6.1.162 BUTTON4_DOUBLE_CLICKED	53
5.6.1.163 BUTTON4_PRESSED	53
5.6.1.164 BUTTON4_RELEASED	53
5.6.1.165 BUTTON4_TRIPLE_CLICKED	53
5.6.1.166 BUTTON5_CLICKED	53
5.6.1.167 BUTTON5_DOUBLE_CLICKED	53
5.6.1.168 BUTTON5_PRESSED	53
5.6.1.169 BUTTON5_RELEASED	53
5.6.1.170 BUTTON5_TRIPLE_CLICKED	53
5.6.1.171 BUTTON_ACTION_MASK	53
5.6.1.172 BUTTON_ALT	53
5.6.1.173 BUTTON_CHANGED	53

5.6.1.174	BUTTON_CLICKED	53
5.6.1.175	BUTTON_CONTROL	53
5.6.1.176	BUTTON_DOUBLE_CLICKED	53
5.6.1.177	BUTTON_MODIFIER_ALT	53
5.6.1.178	BUTTON_MODIFIER_CONTROL	53
5.6.1.179	BUTTON_MODIFIER_MASK	53
5.6.1.180	BUTTON_MODIFIER_SHIFT	53
5.6.1.181	BUTTON_MOVED	53
5.6.1.182	BUTTON_PRESSED	53
5.6.1.183	BUTTON_RELEASED	53
5.6.1.184	BUTTON_SHIFT	53
5.6.1.185	BUTTON_STATUS	53
5.6.1.186	BUTTON_TRIPLE_CLICKED	53
5.6.1.187	CHR_MSK	53
5.6.1.188	CHTYPE_LONG	53
5.6.1.189	COLOR_BLACK	54
5.6.1.190	COLOR_BLUE	54
5.6.1.191	COLOR_CYAN	54
5.6.1.192	COLOR_GREEN	54
5.6.1.193	COLOR_MAGENTA	54
5.6.1.194	COLOR_PAIR	54
5.6.1.195	COLOR_RED	54
5.6.1.196	COLOR_WHITE	54
5.6.1.197	COLOR_YELLOW	54
5.6.1.198	CTL_BKSP	54
5.6.1.199	CTL_DEL	54
5.6.1.200	CTL_DOWN	54
5.6.1.201	CTL_END	54
5.6.1.202	CTL_ENTER	54
5.6.1.203	CTL_HOME	54
5.6.1.204	CTL_INS	54
5.6.1.205	CTL_LEFT	54
5.6.1.206	CTL_PAD0	54
5.6.1.207	CTL_PAD1	54
5.6.1.208	CTL_PAD2	54
5.6.1.209	CTL_PAD3	54
5.6.1.210	CTL_PAD4	54
5.6.1.211	CTL_PAD5	54
5.6.1.212	CTL_PAD6	54
5.6.1.213	CTL_PAD7	54

5.6.1.214 CTL_PAD8	54
5.6.1.215 CTL_PAD9	54
5.6.1.216 CTL_PADCENTER	54
5.6.1.217 CTL_PADENTER	55
5.6.1.218 CTL_PADMINUS	55
5.6.1.219 CTL_PADPLUS	55
5.6.1.220 CTL_PADSLASH	55
5.6.1.221 CTL_PADSTAR	55
5.6.1.222 CTL_PADSTOP	55
5.6.1.223 CTL_PGDN	55
5.6.1.224 CTL_PGUP	55
5.6.1.225 CTL_RIGHT	55
5.6.1.226 CTL_TAB	55
5.6.1.227 CTL_UP	55
5.6.1.228 ERR	55
5.6.1.229 FALSE	55
5.6.1.230 getbegyx	55
5.6.1.231 getch	55
5.6.1.232 getmaxyx	55
5.6.1.233 getparyx	55
5.6.1.234 getsyx	55
5.6.1.235 getyx	55
5.6.1.236 KEY_A1	55
5.6.1.237 KEY_A2	55
5.6.1.238 KEY_A3	55
5.6.1.239 KEY_ABORT	55
5.6.1.240 KEY_ALT_L	55
5.6.1.241 KEY_ALT_R	55
5.6.1.242 KEY_B1	56
5.6.1.243 KEY_B2	56
5.6.1.244 KEY_B3	56
5.6.1.245 KEY_BACKSPACE	56
5.6.1.246 KEY_BEG	56
5.6.1.247 KEY_BREAK	56
5.6.1.248 KEY_BTAB	56
5.6.1.249 KEY_C1	56
5.6.1.250 KEY_C2	56
5.6.1.251 KEY_C3	56
5.6.1.252 KEY_CANCEL	56
5.6.1.253 KEY_CATAB	56

5.6.1.254 KEY_CLEAR	56
5.6.1.255 KEY_CLOSE	56
5.6.1.256 KEY_CODE_YES	56
5.6.1.257 KEY_COMMAND	56
5.6.1.258 KEY_CONTROL_L	56
5.6.1.259 KEY_CONTROL_R	56
5.6.1.260 KEY_COPY	56
5.6.1.261 KEY_CREATE	56
5.6.1.262 KEY_CTAB	56
5.6.1.263 KEY_DC	56
5.6.1.264 KEY_DL	56
5.6.1.265 KEY_DOWN	56
5.6.1.266 KEY_EIC	56
5.6.1.267 KEY_END	56
5.6.1.268 KEY_ENTER	56
5.6.1.269 KEY_EOL	56
5.6.1.270 KEY_EOS	57
5.6.1.271 KEY_EXIT	57
5.6.1.272 KEY_F	57
5.6.1.273 KEY_F0	57
5.6.1.274 KEY_FIND	57
5.6.1.275 KEY_HELP	57
5.6.1.276 KEY_HOME	57
5.6.1.277 KEY_IC	57
5.6.1.278 KEY_IL	57
5.6.1.279 KEY_LEFT	57
5.6.1.280 KEY_LHELP	57
5.6.1.281 KEY_LL	57
5.6.1.282 KEY_MARK	57
5.6.1.283 KEY_MAX	57
5.6.1.284 KEY_MESSAGE	57
5.6.1.285 KEY_MIN	57
5.6.1.286 KEY_MOUSE	57
5.6.1.287 KEY_MOVE	57
5.6.1.288 KEY_NEXT	57
5.6.1.289 KEY_NPAGE	57
5.6.1.290 KEY_OPEN	57
5.6.1.291 KEY_OPTIONS	57
5.6.1.292 KEY_PPAGE	57
5.6.1.293 KEY_PREVIOUS	57

5.6.1.294 KEY_PRINT	57
5.6.1.295 KEY_REDO	57
5.6.1.296 KEY_REFERENCE	57
5.6.1.297 KEY_REFRESH	57
5.6.1.298 KEY_REPLACE	58
5.6.1.299 KEY_RESET	58
5.6.1.300 KEY_RESIZE	58
5.6.1.301 KEY_RESTART	58
5.6.1.302 KEY_RESUME	58
5.6.1.303 KEY_RIGHT	58
5.6.1.304 KEY_SAVE	58
5.6.1.305 KEY_SBEG	58
5.6.1.306 KEY_SCANCEL	58
5.6.1.307 KEY_SCOMMAND	58
5.6.1.308 KEY_SCOPY	58
5.6.1.309 KEY_SCREATE	58
5.6.1.310 KEY_SDC	58
5.6.1.311 KEY_SDL	58
5.6.1.312 KEY_SDOWN	58
5.6.1.313 KEY_SELECT	58
5.6.1.314 KEY_SEND	58
5.6.1.315 KEY_SEOL	58
5.6.1.316 KEY_SEXIT	58
5.6.1.317 KEY_SF	58
5.6.1.318 KEY_SFIND	58
5.6.1.319 KEY_SHELP	58
5.6.1.320 KEY_SHIFT_L	58
5.6.1.321 KEY_SHIFT_R	58
5.6.1.322 KEY_SHOME	58
5.6.1.323 KEY_SIC	58
5.6.1.324 KEY_SLEFT	58
5.6.1.325 KEY_SMESSAGE	58
5.6.1.326 KEY_SMOVE	59
5.6.1.327 KEY_SNEXT	59
5.6.1.328 KEY_SOPTIONS	59
5.6.1.329 KEY_SPREVIOUS	59
5.6.1.330 KEY_SPRINT	59
5.6.1.331 KEY_SR	59
5.6.1.332 KEY_SREDO	59
5.6.1.333 KEY_SREPLACE	59

5.6.1.334 KEY_SRESET	59
5.6.1.335 KEY_SRIGHT	59
5.6.1.336 KEY_SRSUME	59
5.6.1.337 KEY_SSAVE	59
5.6.1.338 KEY_SSUSPEND	59
5.6.1.339 KEY_STAB	59
5.6.1.340 KEY_SUNDO	59
5.6.1.341 KEY_SUP	59
5.6.1.342 KEY_SUSPEND	59
5.6.1.343 KEY_UNDO	59
5.6.1.344 KEY_UP	59
5.6.1.345 MOUSE_MOVED	59
5.6.1.346 MOUSE_POS_REPORT	59
5.6.1.347 MOUSE_WHEEL_DOWN	59
5.6.1.348 MOUSE_WHEEL_SCROLL	59
5.6.1.349 MOUSE_WHEEL_UP	59
5.6.1.350 MOUSE_X_POS	59
5.6.1.351 MOUSE_Y_POS	59
5.6.1.352 NULL	59
5.6.1.353 OK	59
5.6.1.354 PAD0	60
5.6.1.355 PADENTER	60
5.6.1.356 PADMINUS	60
5.6.1.357 PADPLUS	60
5.6.1.358 PADSLASH	60
5.6.1.359 PADSTAR	60
5.6.1.360 PADSTOP	60
5.6.1.361 PAIR_NUMBER	60
5.6.1.362 PDC_ATTR_SHIFT	60
5.6.1.363 PDC_BUILD	60
5.6.1.364 PDC_BUTTON_ALT	60
5.6.1.365 PDC_BUTTON_CONTROL	60
5.6.1.366 PDC_BUTTON_SHIFT	60
5.6.1.367 PDC_CLIP_ACCESS_ERROR	60
5.6.1.368 PDC_CLIP_EMPTY	60
5.6.1.369 PDC_CLIP_MEMORY_ERROR	60
5.6.1.370 PDC_CLIP_SUCCESS	60
5.6.1.371 PDC_COLOR_SHIFT	60
5.6.1.372 PDC_KEY_MODIFIER_ALT	60
5.6.1.373 PDC_KEY_MODIFIER_CONTROL	60

5.6.1.374 PDC_KEY_MODIFIER_NUMLOCK	60
5.6.1.375 PDC_KEY_MODIFIER_SHIFT	60
5.6.1.376 PDC_MOUSE_MOVED	60
5.6.1.377 PDC_MOUSE_POSITION	60
5.6.1.378 PDC_MOUSE_WHEEL_DOWN	60
5.6.1.379 PDC_MOUSE_WHEEL_UP	60
5.6.1.380 PDCEX	60
5.6.1.381 PDCURSES	60
5.6.1.382 REPORT_MOUSE_POSITION	61
5.6.1.383 SHF_DC	61
5.6.1.384 SHF_DOWN	61
5.6.1.385 SHF_IC	61
5.6.1.386 SHF_PADENTER	61
5.6.1.387 SHF_PADMINUS	61
5.6.1.388 SHF_PADPLUS	61
5.6.1.389 SHF_PADSLASH	61
5.6.1.390 SHF_PADSTAR	61
5.6.1.391 SHF_UP	61
5.6.1.392 SYSVcurses	61
5.6.1.393 TRUE	61
5.6.1.394 ungetch	61
5.6.1.395 WA_ALTCHARSET	61
5.6.1.396 WA_BLINK	61
5.6.1.397 WA_BOLD	61
5.6.1.398 WA_DIM	61
5.6.1.399 WA_HORIZONTAL	61
5.6.1.400 WA_INVIS	61
5.6.1.401 WA_LEFT	61
5.6.1.402 WA_LOW	61
5.6.1.403 WA_PROTECT	61
5.6.1.404 WA_REVERSE	61
5.6.1.405 WA_RIGHT	61
5.6.1.406 WA_STANDOUT	61
5.6.1.407 WA_TOP	61
5.6.1.408 WA_UNDERLINE	61
5.6.1.409 WA_VERTICAL	61
5.6.1.410 WHEEL_SCROLLED	62
5.6.1.411 XOPEN	62
5.6.2 Documentación de los 'typedefs'	62
5.6.2.1 attr_t	62

5.6.2.2	bool	62
5.6.2.3	chtype	62
5.6.2.4	mmask_t	62
5.6.2.5	WINDOW	62
5.6.3	Documentación de las funciones	62
5.6.3.1	addch(const chtype)	62
5.6.3.2	addchnstr(const chtype *, int)	62
5.6.3.3	addchstr(const chtype *)	62
5.6.3.4	addnstr(const char *, int)	62
5.6.3.5	addrawch(chtype)	62
5.6.3.6	addstr(const char *)	62
5.6.3.7	assume_default_colors(int, int)	62
5.6.3.8	attr_get(attr_t *, short *, void *)	62
5.6.3.9	attr_off(attr_t, void *)	62
5.6.3.10	attr_on(attr_t, void *)	62
5.6.3.11	attr_set(attr_t, short, void *)	62
5.6.3.12	attroff(chtype)	62
5.6.3.13	attron(chtype)	62
5.6.3.14	attrset(chtype)	62
5.6.3.15	baudrate(void)	62
5.6.3.16	beep(void)	62
5.6.3.17	bkgd(chtype)	62
5.6.3.18	bkgdset(chtype)	62
5.6.3.19	border(chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype)	62
5.6.3.20	box(WINDOW *, chtype, chtype)	63
5.6.3.21	can_change_color(void)	63
5.6.3.22	cbreak(void)	63
5.6.3.23	chgat(int, attr_t, short, const void *)	63
5.6.3.24	clear(void)	63
5.6.3.25	clearok(WINDOW *, bool)	63
5.6.3.26	clrtoebot(void)	63
5.6.3.27	clrtoeol(void)	63
5.6.3.28	color_content(short, short *, short *, short *)	63
5.6.3.29	color_set(short, void *)	63
5.6.3.30	copywin(const WINDOW *, WINDOW *, int, int, int, int, int, int, int)	63
5.6.3.31	crmode(void)	63
5.6.3.32	curs_set(int)	63
5.6.3.33	curses_version(void)	63
5.6.3.34	def_prog_mode(void)	63
5.6.3.35	def_shell_mode(void)	63

5.6.3.36	delay_output(int)	63
5.6.3.37	delch(void)	63
5.6.3.38	deleteln(void)	63
5.6.3.39	delscreen(SCREEN *)	63
5.6.3.40	delwin(WINDOW *)	63
5.6.3.41	derwin(WINDOW *, int, int, int, int)	63
5.6.3.42	doupdate(void)	63
5.6.3.43	draino(int)	63
5.6.3.44	dupwin(WINDOW *)	63
5.6.3.45	echo(void)	63
5.6.3.46	echochar(const chtype)	63
5.6.3.47	endwin(void)	63
5.6.3.48	erase(void)	64
5.6.3.49	erasechar(void)	64
5.6.3.50	filter(void)	64
5.6.3.51	fixterm(void)	64
5.6.3.52	flash(void)	64
5.6.3.53	flushinp(void)	64
5.6.3.54	getattrs(WINDOW *)	64
5.6.3.55	getbegx(WINDOW *)	64
5.6.3.56	getbegy(WINDOW *)	64
5.6.3.57	getbkgd(WINDOW *)	64
5.6.3.58	getbmap(void)	64
5.6.3.59	getcurx(WINDOW *)	64
5.6.3.60	getcury(WINDOW *)	64
5.6.3.61	getmaxx(WINDOW *)	64
5.6.3.62	getmaxy(WINDOW *)	64
5.6.3.63	getmouse(void)	64
5.6.3.64	getnstr(char *, int)	64
5.6.3.65	getparx(WINDOW *)	64
5.6.3.66	getpary(WINDOW *)	64
5.6.3.67	getstr(char *)	64
5.6.3.68	getwin(FILE *)	64
5.6.3.69	halfdelay(int)	64
5.6.3.70	has_colors(void)	64
5.6.3.71	has_ic(void)	64
5.6.3.72	has_il(void)	64
5.6.3.73	has_key(int)	64
5.6.3.74	hline(chtype, int)	64
5.6.3.75	idcok(WINDOW *, bool)	64

5.6.3.76	idlok(WINDOW *, bool)	65
5.6.3.77	immedok(WINDOW *, bool)	65
5.6.3.78	inch(void)	65
5.6.3.79	inchnstr(chtype *, int)	65
5.6.3.80	inchstr(chtype *)	65
5.6.3.81	init_color(short, short, short, short)	65
5.6.3.82	init_pair(short, short, short)	65
5.6.3.83	initscr(void)	65
5.6.3.84	innstr(char *, int)	65
5.6.3.85	insch(chtype)	65
5.6.3.86	insdelln(int)	65
5.6.3.87	insertln(void)	65
5.6.3.88	insnstr(const char *, int)	65
5.6.3.89	insrawch(chtype)	65
5.6.3.90	insstr(const char *)	65
5.6.3.91	instr(char *)	65
5.6.3.92	intrflush(WINDOW *, bool)	65
5.6.3.93	is_linetouched(WINDOW *, int)	65
5.6.3.94	is_termresized(void)	65
5.6.3.95	is_wintouched(WINDOW *)	65
5.6.3.96	isendwin(void)	65
5.6.3.97	keyname(int)	65
5.6.3.98	keypad(WINDOW *, bool)	65
5.6.3.99	killchar(void)	65
5.6.3.100	leaveok(WINDOW *, bool)	65
5.6.3.101	longname(void)	65
5.6.3.102	map_button(unsigned long)	65
5.6.3.103	meta(WINDOW *, bool)	65
5.6.3.104	mouse_off(unsigned long)	66
5.6.3.105	mouse_on(unsigned long)	66
5.6.3.106	mouse_set(unsigned long)	66
5.6.3.107	mouse_trafo(int *, int *, bool)	66
5.6.3.108	mouseinterval(int)	66
5.6.3.109	mousemask(mmask_t, mmask_t *)	66
5.6.3.110	move(int, int)	66
5.6.3.111	mvaddch(int, int, const chtype)	66
5.6.3.112	mvaddchnstr(int, int, const chtype *, int)	66
5.6.3.113	mvaddchstr(int, int, const chtype *)	66
5.6.3.114	mvaddnstr(int, int, const char *, int)	66
5.6.3.115	mvaddrawch(int, int, chtype)	66

5.6.3.116 mvaddstr(int, int, const char *)	66
5.6.3.117 mvchgat(int, int, int, attr_t, short, const void *)	66
5.6.3.118 mvcur(int, int, int, int)	66
5.6.3.119 mvdelch(int, int)	66
5.6.3.120 mvdeleteln(int, int)	66
5.6.3.121 mvderwin(WINDOW *, int, int)	66
5.6.3.122 mvgetch(int, int)	66
5.6.3.123 mvgetnstr(int, int, char *, int)	66
5.6.3.124 mvgetstr(int, int, char *)	66
5.6.3.125 mvhline(int, int, chtype, int)	66
5.6.3.126 mvinch(int, int)	66
5.6.3.127 mvinchnstr(int, int, chtype *, int)	66
5.6.3.128 mvinchstr(int, int, chtype *)	66
5.6.3.129 mvinnstr(int, int, char *, int)	66
5.6.3.130 mvinsch(int, int, chtype)	66
5.6.3.131 mvinsertln(int, int)	66
5.6.3.132 mvinsnstr(int, int, const char *, int)	67
5.6.3.133 mvinsrawch(int, int, chtype)	67
5.6.3.134 mvinsstr(int, int, const char *)	67
5.6.3.135 mvinstr(int, int, char *)	67
5.6.3.136 mvprintw(int, int, const char *,...)	67
5.6.3.137 mvscanw(int, int, const char *,...)	67
5.6.3.138 mvvline(int, int, chtype, int)	67
5.6.3.139 mvwaddch(WINDOW *, int, int, const chtype)	67
5.6.3.140 mvwaddchnstr(WINDOW *, int, int, const chtype *, int)	67
5.6.3.141 mvwaddchstr(WINDOW *, int, int, const chtype *)	67
5.6.3.142 mvwaddnstr(WINDOW *, int, int, const char *, int)	67
5.6.3.143 mvwaddrawch(WINDOW *, int, int, chtype)	67
5.6.3.144 mvwaddstr(WINDOW *, int, int, const char *)	67
5.6.3.145 mvwchgat(WINDOW *, int, int, int, attr_t, short, const void *)	67
5.6.3.146 mvwdelch(WINDOW *, int, int)	67
5.6.3.147 mvwdeleteln(WINDOW *, int, int)	67
5.6.3.148 mvwgetch(WINDOW *, int, int)	67
5.6.3.149 mvwgetnstr(WINDOW *, int, int, char *, int)	67
5.6.3.150 mvwgetstr(WINDOW *, int, int, char *)	67
5.6.3.151 mvwhline(WINDOW *, int, int, chtype, int)	67
5.6.3.152 mvwin(WINDOW *, int, int)	67
5.6.3.153 mvwinch(WINDOW *, int, int)	67
5.6.3.154 mvwinchnstr(WINDOW *, int, int, chtype *, int)	67
5.6.3.155 mvwinchstr(WINDOW *, int, int, chtype *)	67

5.6.3.156 mvwinstr(WINDOW *, int, int, char *, int)	67
5.6.3.157 mvwinsch(WINDOW *, int, int, chtype)	67
5.6.3.158 mvwinstrln(WINDOW *, int, int)	67
5.6.3.159 mvwinsnstr(WINDOW *, int, int, const char *, int)	67
5.6.3.160 mvwinsrawch(WINDOW *, int, int, chtype)	68
5.6.3.161 mvwinsstr(WINDOW *, int, int, const char *)	68
5.6.3.162 mvwinstr(WINDOW *, int, int, char *)	68
5.6.3.163 mvwprintw(WINDOW *, int, int, const char *,...)	68
5.6.3.164 mvwscanw(WINDOW *, int, int, const char *,...)	68
5.6.3.165 mvwvline(WINDOW *, int, int, chtype, int)	68
5.6.3.166 napms(int)	68
5.6.3.167 nc_getmouse(MEVENT *)	68
5.6.3.168 newpad(int, int)	68
5.6.3.169 newterm(const char *, FILE *, FILE *)	68
5.6.3.170 newwin(int, int, int, int)	68
5.6.3.171 nl(void)	68
5.6.3.172 nocbreak(void)	68
5.6.3.173 nocrmode(void)	68
5.6.3.174 nodelay(WINDOW *, bool)	68
5.6.3.175 noecho(void)	68
5.6.3.176 nonl(void)	68
5.6.3.177 noqiflush(void)	68
5.6.3.178 noraw(void)	68
5.6.3.179 notimeout(WINDOW *, bool)	68
5.6.3.180 overlay(const WINDOW *, WINDOW *)	68
5.6.3.181 overwrite(const WINDOW *, WINDOW *)	68
5.6.3.182 pair_content(short, short *, short *)	68
5.6.3.183 PDC_clearclipboard(void)	68
5.6.3.184 PDC_debug(const char *,...)	68
5.6.3.185 PDC_freeclipboard(char *)	68
5.6.3.186 PDC_get_input_fd(void)	68
5.6.3.187 PDC_get_key_modifiers(void)	68
5.6.3.188 PDC_getclipboard(char **, long *)	69
5.6.3.189 PDC_return_key_modifiers(bool)	69
5.6.3.190 PDC_save_key_modifiers(bool)	69
5.6.3.191 PDC_set_blink(bool)	69
5.6.3.192 PDC_set_line_color(short)	69
5.6.3.193 PDC_set_title(const char *)	69
5.6.3.194 PDC_setclipboard(const char *, long)	69
5.6.3.195 PDC_ungetch(int)	69

5.6.3.196 pechochar(WINDOW *, chtype)	69
5.6.3.197 pnoutrefresh(WINDOW *, int, int, int, int, int, int)	69
5.6.3.198 prefresh(WINDOW *, int, int, int, int, int, int)	69
5.6.3.199 printw(const char *,...)	69
5.6.3.200 putwin(WINDOW *, FILE *)	69
5.6.3.201 qiflush(void)	69
5.6.3.202 raw(void)	69
5.6.3.203 raw_output(bool)	69
5.6.3.204 redrawwin(WINDOW *)	69
5.6.3.205 refresh(void)	69
5.6.3.206 request_mouse_pos(void)	69
5.6.3.207 reset_prog_mode(void)	69
5.6.3.208 reset_shell_mode(void)	69
5.6.3.209 resetterm(void)	69
5.6.3.210 resetty(void)	69
5.6.3.211 resize_term(int, int)	69
5.6.3.212 resize_window(WINDOW *, int, int)	69
5.6.3.213 ripoffline(int, int*)(WINDOW *, int)	69
5.6.3.214 saveterm(void)	69
5.6.3.215 savetty(void)	69
5.6.3.216 scanw(const char *,...)	70
5.6.3.217 scr_dump(const char *)	70
5.6.3.218 scr_init(const char *)	70
5.6.3.219 scr_restore(const char *)	70
5.6.3.220 scr_set(const char *)	70
5.6.3.221 scr1(int)	70
5.6.3.222 scroll(WINDOW *)	70
5.6.3.223 scrollok(WINDOW *, bool)	70
5.6.3.224 set_term(SCREEN *)	70
5.6.3.225 setscrreg(int, int)	70
5.6.3.226 setsyx(int, int)	70
5.6.3.227 slk_attr_off(const attr_t, void *)	70
5.6.3.228 slk_attr_on(const attr_t, void *)	70
5.6.3.229 slk_attr_set(const attr_t, short, void *)	70
5.6.3.230 slk_attroff(const chtype)	70
5.6.3.231 slk_attron(const chtype)	70
5.6.3.232 slk_attrset(const chtype)	70
5.6.3.233 slk_clear(void)	70
5.6.3.234 slk_color(short)	70
5.6.3.235 slk_init(int)	70

5.6.3.236 slk_label(int)	70
5.6.3.237 slk_noutrefresh(void)	70
5.6.3.238 slk_refresh(void)	70
5.6.3.239 slk_restore(void)	70
5.6.3.240 slk_set(int, const char *, int)	70
5.6.3.241 slk_touch(void)	70
5.6.3.242 standend(void)	70
5.6.3.243 standout(void)	70
5.6.3.244 start_color(void)	71
5.6.3.245 subpad(WINDOW *, int, int, int, int)	71
5.6.3.246 subwin(WINDOW *, int, int, int, int)	71
5.6.3.247 syncok(WINDOW *, bool)	71
5.6.3.248 term_attrs(void)	71
5.6.3.249 termattrs(void)	71
5.6.3.250 termname(void)	71
5.6.3.251 timeout(int)	71
5.6.3.252 touchline(WINDOW *, int, int)	71
5.6.3.253 touchwin(WINDOW *)	71
5.6.3.254 traceoff(void)	71
5.6.3.255 traceon(void)	71
5.6.3.256 typeahead(int)	71
5.6.3.257 unctrl(chtype)	71
5.6.3.258 ungetmouse(MEVENT *)	71
5.6.3.259 untouchwin(WINDOW *)	71
5.6.3.260 use_default_colors(void)	71
5.6.3.261 use_env(bool)	71
5.6.3.262 vid_attr(attr_t, short, void *)	71
5.6.3.263 vid_puts(attr_t, short, void *, int(*) (int))	71
5.6.3.264 vidattr(chtype)	71
5.6.3.265 vidputs(chtype, int(*) (int))	71
5.6.3.266 vline(chtype, int)	71
5.6.3.267 vw_printw(WINDOW *, const char *, va_list)	71
5.6.3.268 vw_scanw(WINDOW *, const char *, va_list)	71
5.6.3.269 vwprintw(WINDOW *, const char *, va_list)	71
5.6.3.270 vwscanw(WINDOW *, const char *, va_list)	71
5.6.3.271 waddch(WINDOW *, const chtype)	71
5.6.3.272 waddchnstr(WINDOW *, const chtype *, int)	72
5.6.3.273 waddchstr(WINDOW *, const chtype *)	72
5.6.3.274 waddnstr(WINDOW *, const char *, int)	72
5.6.3.275 waddrawch(WINDOW *, chtype)	72

5.6.3.276 waddstr(WINDOW *, const char *)	72
5.6.3.277 wattr_get(WINDOW *, attr_t *, short *, void *)	72
5.6.3.278 wattr_off(WINDOW *, attr_t, void *)	72
5.6.3.279 wattr_on(WINDOW *, attr_t, void *)	72
5.6.3.280 wattr_set(WINDOW *, attr_t, short, void *)	72
5.6.3.281 wattroff(WINDOW *, chtype)	72
5.6.3.282 wattron(WINDOW *, chtype)	72
5.6.3.283 wattrset(WINDOW *, chtype)	72
5.6.3.284 wbkgd(WINDOW *, chtype)	72
5.6.3.285 wbkgdset(WINDOW *, chtype)	72
5.6.3.286 wborder(WINDOW *, chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype)	72
5.6.3.287 wchgat(WINDOW *, int, attr_t, short, const void *)	72
5.6.3.288 wclear(WINDOW *)	72
5.6.3.289 wclrtobot(WINDOW *)	72
5.6.3.290 wclrtoeol(WINDOW *)	72
5.6.3.291 wcolor_set(WINDOW *, short, void *)	72
5.6.3.292 wcursyncup(WINDOW *)	72
5.6.3.293 wdelch(WINDOW *)	72
5.6.3.294 wdeleteln(WINDOW *)	72
5.6.3.295 wechochar(WINDOW *, const chtype)	72
5.6.3.296 wenclose(const WINDOW *, int, int)	72
5.6.3.297 werase(WINDOW *)	72
5.6.3.298 wgetch(WINDOW *)	72
5.6.3.299 wgetnstr(WINDOW *, char *, int)	72
5.6.3.300 wgetstr(WINDOW *, char *)	73
5.6.3.301 whline(WINDOW *, chtype, int)	73
5.6.3.302 winch(WINDOW *)	73
5.6.3.303 winchnstr(WINDOW *, chtype *, int)	73
5.6.3.304 winchstr(WINDOW *, chtype *)	73
5.6.3.305 winnstr(WINDOW *, char *, int)	73
5.6.3.306 winsch(WINDOW *, chtype)	73
5.6.3.307 winsdelln(WINDOW *, int)	73
5.6.3.308 winsertln(WINDOW *)	73
5.6.3.309 winsnstr(WINDOW *, const char *, int)	73
5.6.3.310 winsrawch(WINDOW *, chtype)	73
5.6.3.311 winsstr(WINDOW *, const char *)	73
5.6.3.312 winstr(WINDOW *, char *)	73
5.6.3.313 wmouse_position(WINDOW *, int *, int *)	73
5.6.3.314 wmouse_trafo(const WINDOW *, int *, int *, bool)	73

5.6.3.315	wmove(WINDOW *, int, int)	73
5.6.3.316	wnoutrefresh(WINDOW *)	73
5.6.3.317	wordchar(void)	73
5.6.3.318	wprintw(WINDOW *, const char *,...)	73
5.6.3.319	wredrawln(WINDOW *, int, int)	73
5.6.3.320	wrefresh(WINDOW *)	73
5.6.3.321	wresize(WINDOW *, int, int)	73
5.6.3.322	wscanw(WINDOW *, const char *,...)	73
5.6.3.323	wscrln(WINDOW *, int)	73
5.6.3.324	wsetscrreg(WINDOW *, int, int)	73
5.6.3.325	wstandend(WINDOW *)	73
5.6.3.326	wstandout(WINDOW *)	73
5.6.3.327	wsyncdown(WINDOW *)	73
5.6.3.328	wsyncup(WINDOW *)	74
5.6.3.329	wtimeout(WINDOW *, int)	74
5.6.3.330	wtouchln(WINDOW *, int, int, int)	74
5.6.3.331	wvline(WINDOW *, chtype, int)	74
5.6.4	Documentación de las variables	74
5.6.4.1	acs_map	74
5.6.4.2	COLOR_PAIRS	74
5.6.4.3	COLORS	74
5.6.4.4	COLS	74
5.6.4.5	curscr	74
5.6.4.6	LINES	74
5.6.4.7	Mouse_status	74
5.6.4.8	SP	74
5.6.4.9	stdscr	74
5.6.4.10	TABSIZE	74
5.6.4.11	ttytype	74
5.7	Referencia del Archivo decoder.c	74
5.7.1	Documentación de las funciones	75
5.7.1.1	countLines(FILE *fp)	75
5.7.1.2	decodeInstruction(instruction_t instruction, uint32_t *Registro, char *R_Banderas)	75
5.7.1.3	getInstruction(char *instStr)	75
5.7.1.4	readFile(char *filename, ins_t *instructions)	75
5.8	Referencia del Archivo decoder.h	75
5.8.1	Documentación de las funciones	76
5.8.1.1	countLines(FILE *fp)	76
5.8.1.2	decodeInstruction(instruction_t instruction, uint32_t *Registro, char *R_Banderas)	76
5.8.1.3	getInstruction(char *instStr)	77

5.8.1.4	readFile(char *filename, ins_t *instructions)	77
5.9	Referencia del Archivo desplazamiento.c	77
5.9.1	Documentación de los 'defines'	78
5.9.1.1	C	78
5.9.1.2	PC	78
5.9.2	Documentación de las funciones	78
5.9.2.1	ASR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↔ Banderas)	78
5.9.2.2	BIC(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)	78
5.9.2.3	LSL(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	78
5.9.2.4	LSR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↔ Banderas)	79
5.9.2.5	MVN(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)	79
5.9.2.6	NOP(uint32_t *Registro)	79
5.9.2.7	REV(uint32_t *Registro, uint32_t *Rd)	79
5.9.2.8	REVIG(uint32_t *Registro, uint32_t *Rd)	79
5.9.2.9	REVSH(uint32_t *Registro, uint32_t *Rd)	80
5.9.2.10	ROR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↔ Banderas)	80
5.9.2.11	RSB(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, int inmediato, char *R_↔ Banderas)	80
5.10	Referencia del Archivo desplazamiento.h	80
5.10.1	Documentación de las funciones	81
5.10.1.1	ASR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↔ Banderas)	81
5.10.1.2	BIC(uint32_t *Registro, uint32_t *Rd, uint32_t Rm, char *R_Banderas)	81
5.10.1.3	LSL(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)	81
5.10.1.4	LSR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↔ Banderas)	82
5.10.1.5	MVN(uint32_t *Registro, uint32_t *Rd, uint32_t Rm, char *R_Banderas)	82
5.10.1.6	NOP(uint32_t *Registro)	82
5.10.1.7	REV(uint32_t *Registro, uint32_t *Rd)	82
5.10.1.8	REVIG(uint32_t *Registro, uint32_t *Rd)	82
5.10.1.9	REVSH(uint32_t *Registro, uint32_t *Rd)	83
5.10.1.10	ROR(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_↔ Banderas)	83
5.10.1.11	RSB(uint32_t *Registro, uint32_t *Rd, uint32_t Rn, int inmediato, char *R_↔ Banderas)	83
5.11	Referencia del Archivo main.c	83
5.11.1	Documentación de los 'defines'	84
5.11.1.1	PC	84
5.11.2	Documentación de las funciones	84

5.11.2.1	main()	84
5.12	Referencia del Archivo mostrar.c	84
5.12.1	Documentación de los 'defines'	84
5.12.1.1	C	84
5.12.1.2	LR	84
5.12.1.3	N	84
5.12.1.4	PC	84
5.12.1.5	V	84
5.12.1.6	Z	85
5.12.2	Documentación de las funciones	85
5.12.2.1	mostrar_valores(uint32_t *Registro, char *R_Banderas)	85
5.13	Referencia del Archivo mostrar.h	85
5.13.1	Documentación de las funciones	85
5.13.1.1	mostrar_valores(uint32_t *Registro, char *R_Banderas)	85
5.14	Referencia del Archivo README.md	85
5.15	Referencia del Archivo test.c	85
5.15.1	Documentación de los 'defines'	86
5.15.1.1	C	86
5.15.1.2	LR	86
5.15.1.3	N	86
5.15.1.4	PC	86
5.15.1.5	V	86
5.15.1.6	Z	86
5.15.2	Documentación de las funciones	86
5.15.2.1	main()	86

Capítulo 1

Trabajo_1_Micros

Cesar Andres Tejada TorresCodigo: 1094950939 Celular: 3147405228

Brayan Andres Gonzalez PotesCodigo: 1094931216 Celular: 3225435194

Juan David Aragon JaramilloCodigo: 1115191534 Celular: 3104728273

Capítulo 2

Índice de estructura de datos

2.1. Estructura de datos

Lista de estructuras con una breve descripción:

_win	7
ins_t	8
instruction_t	9
MEVENT	9
MOUSE_STATUS	10
SCREEN	10

Capítulo 3

Indice de archivos

3.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

ALU.c	13
ALU.h	18
branch.c	22
branch.h	27
colors.h	31
curses.h	32
decoder.c	74
decoder.h	75
desplazamiento.c	77
desplazamiento.h	80
main.c	83
mostrar.c	84
mostrar.h	85
test.c	85

Capítulo 4

Documentación de las estructuras de datos

4.1. Referencia de la Estructura `_win`

```
#include <curses.h>
```

Campos de datos

- `int _cury`
- `int _curx`
- `int _maxy`
- `int _maxx`
- `int _begy`
- `int _begx`
- `int _flags`
- `chtype _attrs`
- `chtype _bkgd`
- `bool _clear`
- `bool _leaveit`
- `bool _scroll`
- `bool _nodelay`
- `bool _immed`
- `bool _sync`
- `bool _use_keypad`
- `chtype ** _y`
- `int * _firstch`
- `int * _lastch`
- `int _tmarg`
- `int _bmarg`
- `int _delayms`
- `int _parx`
- `int _pary`
- `struct _win * _parent`

4.1.1. Documentación de los campos

4.1.1.1. `chtype _attrs`

4.1.1.2. `int _begx`

- 4.1.1.3. `int_begy`
- 4.1.1.4. `chtype_bkgd`
- 4.1.1.5. `int_bmarg`
- 4.1.1.6. `bool_clear`
- 4.1.1.7. `int_curx`
- 4.1.1.8. `int_cury`
- 4.1.1.9. `int_delayms`
- 4.1.1.10. `int* _firstch`
- 4.1.1.11. `int_flags`
- 4.1.1.12. `bool_immed`
- 4.1.1.13. `int* _lastch`
- 4.1.1.14. `bool_leaveit`
- 4.1.1.15. `int_maxx`
- 4.1.1.16. `int_maxy`
- 4.1.1.17. `bool_nodelay`
- 4.1.1.18. `struct_win* _parent`
- 4.1.1.19. `int_parx`
- 4.1.1.20. `int_pary`
- 4.1.1.21. `bool_scroll`
- 4.1.1.22. `bool_sync`
- 4.1.1.23. `int_tmarg`
- 4.1.1.24. `bool_use_keypad`
- 4.1.1.25. `chtype** _y`

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [curses.h](#)

4.2. Referencia de la Estructura `ins_t`

```
#include <decoder.h>
```

Campos de datos

- char ** [array](#)

4.2.1. Documentación de los campos

4.2.1.1. char** array

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [decoder.h](#)

4.3. Referencia de la Estructura instruction_t

```
#include <decoder.h>
```

Campos de datos

- char [mnemonic](#) [10]
- char [op1_type](#)
- char [op2_type](#)
- char [op3_type](#)
- uint32_t [op1_value](#)
- uint32_t [op2_value](#)
- uint32_t [op3_value](#)

4.3.1. Documentación de los campos

4.3.1.1. char mnemonic[10]

4.3.1.2. char op1_type

4.3.1.3. uint32_t op1_value

4.3.1.4. char op2_type

4.3.1.5. uint32_t op2_value

4.3.1.6. char op3_type

4.3.1.7. uint32_t op3_value

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [decoder.h](#)

4.4. Referencia de la Estructura MEVENT

```
#include <curses.h>
```

Campos de datos

- short [id](#)
- int [x](#)
- int [y](#)
- int [z](#)
- [mmask_t](#) [bstate](#)

4.4.1. Documentación de los campos

4.4.1.1. [mmask_t](#) [bstate](#)

4.4.1.2. short [id](#)

4.4.1.3. int [x](#)

4.4.1.4. int [y](#)

4.4.1.5. int [z](#)

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [curses.h](#)

4.5. Referencia de la Estructura `MOUSE_STATUS`

```
#include <curses.h>
```

Campos de datos

- int [x](#)
- int [y](#)
- short [button](#) [3]
- int [changes](#)

4.5.1. Documentación de los campos

4.5.1.1. short [button](#)[3]

4.5.1.2. int [changes](#)

4.5.1.3. int [x](#)

4.5.1.4. int [y](#)

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [curses.h](#)

4.6. Referencia de la Estructura `SCREEN`

```
#include <curses.h>
```


Campos de datos

- `bool alive`
- `bool autocr`
- `bool cbreak`
- `bool echo`
- `bool raw_inp`
- `bool raw_out`
- `bool audible`
- `bool mono`
- `bool resized`
- `bool orig_attr`
- `short orig_fore`
- `short orig_back`
- `int cursrow`
- `int curscol`
- `int visibility`
- `int orig_cursor`
- `int lines`
- `int cols`
- `unsigned long _trap_mbe`
- `unsigned long _map_mbe_to_key`
- `int mouse_wait`
- `int slklines`
- `WINDOW * slk_winptr`
- `int linesrippedoff`
- `int linesrippedoffontop`
- `int delaytenths`
- `bool _preserve`
- `int _restore`
- `bool save_key_modifiers`
- `bool return_key_modifiers`
- `bool key_code`
- `short line_color`

4.6.1. Documentación de los campos

4.6.1.1. `unsigned long _map_mbe_to_key`

4.6.1.2. `bool _preserve`

4.6.1.3. `int _restore`

4.6.1.4. `unsigned long _trap_mbe`

4.6.1.5. `bool alive`

4.6.1.6. `bool audible`

4.6.1.7. `bool autocr`

4.6.1.8. `bool cbreak`

4.6.1.9. `int cols`

- 4.6.1.10. `int curscol`
- 4.6.1.11. `int cursrow`
- 4.6.1.12. `int delaytenths`
- 4.6.1.13. `bool echo`
- 4.6.1.14. `bool key_code`
- 4.6.1.15. `short line_color`
- 4.6.1.16. `int lines`
- 4.6.1.17. `int linesrippedoff`
- 4.6.1.18. `int linesrippedoffontop`
- 4.6.1.19. `bool mono`
- 4.6.1.20. `int mouse_wait`
- 4.6.1.21. `bool orig_attr`
- 4.6.1.22. `short orig_back`
- 4.6.1.23. `int orig_cursor`
- 4.6.1.24. `short orig_fore`
- 4.6.1.25. `bool raw_inp`
- 4.6.1.26. `bool raw_out`
- 4.6.1.27. `bool resized`
- 4.6.1.28. `bool return_key_modifiers`
- 4.6.1.29. `bool save_key_modifiers`
- 4.6.1.30. `WINDOW* slk_winptr`
- 4.6.1.31. `int slklines`
- 4.6.1.32. `int visibility`

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [curses.h](#)

Capítulo 5

Documentación de archivos

5.1. Referencia del Archivo ALU.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "ALU.h"
```

'defines'

- #define N 0
- #define Z 1
- #define C 2
- #define V 3
- #define PC 15

Funciones

- void **Banderas** (uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion para las banderas
- void **Bandera_N** (uint32_t Rd, char *R_Banderas)
Funcion para la bandera N de negativo
- void **Bandera_Z** (uint32_t Rd, char *R_Banderas)
Funcion para la bandera Z de cero
- void **Bandera_C** (uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion para la bandera C de carry
- void **Bandera_V** (uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion para la bandera V de sobreflujo
- void **ADC** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ADC Suma con carry
- void **ADD** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ADD Suma
- void **AND** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion AND Multiplicacion logica
- void **EOR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion EOR Exor
- void **MOV** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)

Funcion MOV Mover un valor de registro a otro

- void **ORR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion ORR Suma logica

- void **SUB** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion SUB Resta

- void **SBC** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion SBC Resta con carry

- void **CMN** (uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion CMN Suma sin resultado, solo modifica banderas

- void **CMP** (uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion CMP Resta sin resultado, solo modifica banderas

- void **MUL** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion MUL Multiplicacion de registros

- void **TST** (uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion TST Multiplicacion logica sin resultado, solo modifica registros**5.1.1. Documentación de los 'defines'**

5.1.1.1. **#define C 2**

5.1.1.2. **#define N 0**

5.1.1.3. **#define PC 15**

5.1.1.4. **#define V 3**

5.1.1.5. **#define Z 1**

5.1.2. Documentación de las funciones

5.1.2.1. void **ADC** (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ADC Suma con carry**Parámetros**

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor de registro
<i>Rm</i>	valor de registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.2. void **ADD** (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ADD Suma**Parámetros**

<i>*Registro</i>	Puntero al registro
------------------	---------------------

<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.3. void AND (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion AND Multiplicacion logica

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.4. void Bandera_C (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion para la bandera C de carry

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>Rn</i>	primer valor de la operacion realizara anteriormente
<i>Rm</i>	segundo valor de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.1.2.5. void Bandera_N (uint32_t *Rd*, char * *R_Banderas*)

Funcion para la bandera N de negativo

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.1.2.6. void Bandera_V (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion para la bandera V de sobreflujo

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>Rn</i>	primer valor de la operacion realizara anteriormente
<i>Rm</i>	segundo valor de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.1.2.7. void Bandera_Z (uint32_t *Rd*, char * *R_Banderas*)

Funcion para la bandera Z de cero

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.1.2.8. void Banderas (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion para las banderas

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>Rn</i>	primer valor de la operacion realizara anteriormente
<i>Rm</i>	segundo valor de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.1.2.9. void CMN (uint32_t * *Registro*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion CMN Suma sin resultado, solo modifica banderas

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rn</i>	Valor del primer registro
<i>Rm</i>	valor del segundo registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.10. void CMP (uint32_t * *Registro*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion CMP Resta sin resultado, solo modifica banderas

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rn</i>	Valor del primer registro
<i>Rm</i>	valor del segundo registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.11. void EOR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion EOR Exor

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro

<i>*R_Banderas</i>	registro de las banderas
--------------------	--------------------------

5.1.2.12. void MOV (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, char * *R_Banderas*)

Funcion MOV Mover un valor de registro a otro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del registro de destino
<i>Rn</i>	valor de registro orgien
<i>*R_Banderas</i>	registro de las banderas

5.1.2.13. void MUL (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion MUL Multiplicacion de registros

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.14. void ORR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ORR Suma logica

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.15. void SBC (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion SBC Resta con carry

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del registro
<i>Rn</i>	valor de registro
<i>Rm</i>	valor de registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.16. void SUB (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion SUB Resta

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.1.2.17. void TST (uint32_t * *Registro*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion TST Multiplicacion logica sin resultado, solo modifica registros

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rn</i>	Valor del registro
<i>Rm</i>	valor del registro
<i>*R_Banderas</i>	registro de las banderas

5.2. Referencia del Archivo ALU.h

```
#include <stdint.h>
```

Funciones

- void **Banderas** (uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion para las banderas
- void **Bandera_N** (uint32_t Rd, char *R_Banderas)
Funcion para la bandera N de negativo
- void **Bandera_Z** (uint32_t Rd, char *R_Banderas)
Funcion para la bandera Z de cero
- void **Bandera_C** (uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion para la bandera C de carry
- void **Bandera_V** (uint32_t Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion para la bandera V de sobreflujo
- void **ADC** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ADC Suma con carry
- void **ADD** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ADD Suma
- void **AND** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion AND Multiplicacion logica
- void **EOR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion EOR Exor
- void **MOV** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)
Funcion MOV Mover un valor de registro a otro
- void **ORR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ORR Suma logica
- void **SUB** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion SUB Resta

- void **SBC** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion SBC Resta con carry

- void **CMN** (uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion CMN Suma sin resultado, solo modifica banderas

- void **CMP** (uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion CMP Resta sin resultado, solo modifica banderas

- void **MUL** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion MUL Multiplicacion de registros

- void **TST** (uint32_t *Registro, uint32_t Rn, uint32_t Rm, char *R_Banderas)

Funcion TST Multiplicacion logica sin resultado, solo modifica registros

5.2.1. Documentación de las funciones

5.2.1.1. void **ADC** (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ADC Suma con carry

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor de registro
<i>Rm</i>	valor de registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.2. void **ADD** (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ADD Suma

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.3. void **AND** (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion AND Multiplicacion logica

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.4. void **Bandera_C** (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion para la bandera C de carry

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>Rn</i>	primer valor de la operacion realizara anteriormente
<i>Rm</i>	segundo valor de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.2.1.5. void Bandera_N (uint32_t *Rd*, char * *R_Banderas*)

Funcion para la bandera N de negativo

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.2.1.6. void Bandera_V (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion para la bandera V de sobreflujo

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>Rn</i>	primer valor de la operacion realizara anteriormente
<i>Rm</i>	segundo valor de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.2.1.7. void Bandera_Z (uint32_t *Rd*, char * *R_Banderas*)

Funcion para la bandera Z de cero

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.2.1.8. void Banderas (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion para las banderas

Parámetros

<i>Rd</i>	valor resultante de la operacion realizada anteriormente
<i>Rn</i>	primer valor de la operacion realizara anteriormente
<i>Rm</i>	segundo valor de la operacion realizada anteriormente
<i>*R_Banderas</i>	registro donde se almacenan las banderas

5.2.1.9. void CMN (uint32_t * *Registro*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion CMN Suma sin resultado, solo modifica banderas

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rn</i>	Valor del primer registro
<i>Rm</i>	valor del segundo registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.10. void CMP (uint32_t * *Registro*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion CMP Resta sin resultado, solo modifica banderas

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rn</i>	Valor del primer registro
<i>Rm</i>	valor del segundo registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.11. void EOR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion EOR Exor

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.12. void MOV (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, char * *R_Banderas*)

Funcion MOV Mover un valor de registro a otro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del registro de destino
<i>Rn</i>	valor de registro orgien
<i>*R_Banderas</i>	registro de las banderas

5.2.1.13. void MUL (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion MUL Multiplicacion de registros

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	puntero del primer registro
<i>Rn</i>	valor del segundo registro
<i>Rm</i>	valor del tercer registro
<i>*R_Banderas</i>	registro de las banderas

5.2.1.14. void ORR (uint32_t * Registro, uint32_t * Rd, uint32_t Rn, uint32_t Rm, char * R_Banderas)

Funcion ORR Suma logica

Parámetros

*Registro	Puntero al registro
*Rd	puntero del primer registro
Rn	valor del segundo registro
Rm	valor del tercer registro
*R_Banderas	registro de las banderas

5.2.1.15. void SBC (uint32_t * Registro, uint32_t * Rd, uint32_t Rn, uint32_t Rm, char * R_Banderas)

Funcion SBC Resta con carry

Parámetros

*Registro	Puntero al registro
*Rd	puntero del registro
Rn	valor de registro
Rm	valor de registro
*R_Banderas	registro de las banderas

5.2.1.16. void SUB (uint32_t * Registro, uint32_t * Rd, uint32_t Rn, uint32_t Rm, char * R_Banderas)

Funcion SUB Resta

Parámetros

*Registro	Puntero al registro
*Rd	puntero del primer registro
Rn	valor del segundo registro
Rm	valor del tercer registro
*R_Banderas	registro de las banderas

5.2.1.17. void TST (uint32_t * Registro, uint32_t Rn, uint32_t Rm, char * R_Banderas)

Funcion TST Multiplicacion logica sin resultado, solo modifica registros

Parámetros

*Registro	Puntero al registro
Rn	Valor del registro
Rm	valor del registro
*R_Banderas	registro de las banderas

5.3. Referencia del Archivo branch.c

```
#include <stdio.h>
#include <stdlib.h>
#include "branch.h"
```

'defines'

- #define N 0
- #define Z 1
- #define C 2
- #define V 3
- #define LR 14
- #define PC 15

Funciones

- void BEQ (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BEQ Salto si es igual
- void BNE (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BNE Salto si no es igual
- void BCS (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BCS Salto si es mayor o igual (sin signo)
- void BCC (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BCC salto si es menor (sin signo)
- void BMI (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BMI Salto si es negativo
- void BPL (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BPL Salto si es positivo
- void BVS (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BVS Salto si hay sobreflujo
- void BVC (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BVC Salto si no hay sobreflujo
- void BHI (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BHI Salto si es mayor (sin signo)
- void BLS (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BLS Salto si es menor o igual (sin signo)
- void BGE (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BGE Salto si es mayor o igual (con signo)
- void BLT (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BLT Salto si es menor (con signo)
- void BGT (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BGT Salto si es mayor (con signo)
- void BLE (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BLE Salto si es menor o igual (con signo)
- void BAL (uint32_t *Registro, uint32_t Label)
BAL Salto sin condicion, de cualquier forma
- void BL (uint32_t *Registro, uint32_t Label)
BL Salto guardando posicion instruccion siguiente en LR
- void BX (uint32_t *Registro)
BX Salta a la posicion guardada en LR
- void B (uint32_t *Registro, uint32_t Label)
B Salto sin condicion, de cualquier forma

5.3.1. Documentación de los 'defines'

5.3.1.1. `#define C 2`

5.3.1.2. `#define LR 14`

5.3.1.3. `#define N 0`

5.3.1.4. `#define PC 15`

5.3.1.5. `#define V 3`

5.3.1.6. `#define Z 1`

5.3.2. Documentación de las funciones

5.3.2.1. `void B (uint32_t * Registro, uint32_t Label)`

B Salto sin condicion, de cualquier forma

Parámetros

<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.2. `void BAL (uint32_t * Registro, uint32_t Label)`

BAL Salto sin condicion, de cualquier forma

Parámetros

<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.3. `void BCC (char * R_Banderas, uint32_t * Registro, uint32_t Label)`

BCC salto si es menor (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.4. `void BCS (char * R_Banderas, uint32_t * Registro, uint32_t Label)`

BCS Salto si es mayor o igual (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.5. void BEQ (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BEQ Salto si es igual

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.6. void BGE (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BGE Salto si es mayor o igual (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.7. void BGT (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BGT Salto si es mayor (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.8. void BHI (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BHI Salto si es mayor (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.9. void BL (uint32_t * *Registro*, uint32_t *Label*)

BL Salto guardando posicion instruccion siguiente en LR

Parámetros

<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.10. void BLE (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BLE Salto si es menor o igual (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.11. void BLS (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BLS Salto si es menor o igual (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.12. void BLT (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BLT Salto si es menor (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.13. void BMI (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BMI Salto si es negativo

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.14. void BNE (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BNE Salto si no es igual

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.15. void BPL (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BPL Salto si es positivo

```
/fn void BPL(char *R_Banderas,uint32_t *Registro,uint32_t Label)
```

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.16. void BVC (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BVC Salto si no hay sobreflujo

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.17. void BVS (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BVS Salto si hay sobreflujo

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.3.2.18. void BX (uint32_t * *Registro*)

BX Salta a la posicion guardada en LR

Parámetros

<i>*Registro</i>	Puntero a el registro
------------------	-----------------------

5.4. Referencia del Archivo branch.h

```
#include <stdint.h>
```

Funciones

- void **BEQ** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BEQ Salto si es igual
- void **BNE** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BNE Salto si no es igual
- void **BCS** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BCS Salto si es mayor o igual (sin signo)
- void **BCC** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BCC salto si es menor (sin signo)
- void **BMI** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BMI Salto si es negativo
- void **BPL** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BPL Salto si es positivo
- void **BVS** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BVS Salto si hay sobreflujo
- void **BVC** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BVC Salto si no hay sobreflujo
- void **BHI** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BHI Salto si es mayor (sin signo)
- void **BLS** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BLS Salto si es menor o igual (sin signo)
- void **BGE** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BGE Salto si es mayor o igual (con signo)
- void **BLT** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BLT Salto si es menor (con signo)
- void **BGT** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BGT Salto si es mayor (con signo)
- void **BLE** (char *R_Banderas, uint32_t *Registro, uint32_t Label)
BLE Salto si es menor o igual (con signo)
- void **BAL** (uint32_t *Registro, uint32_t Label)
BAL Salto sin condicion, de cualquier forma
- void **BL** (uint32_t *Registro, uint32_t Label)
BL Salto guardando posicion instruccion siguiente en LR
- void **BX** (uint32_t *Registro)
BX Salta a la posicion guardada en LR
- void **B** (uint32_t *Registro, uint32_t Label)
B Salto sin condicion, de cualquier forma

5.4.1. Documentación de las funciones

5.4.1.1. void B (uint32_t * Registro, uint32_t Label)

B Salto sin condicion, de cualquier forma

Parámetros

<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.2. void BAL (uint32_t * Registro, uint32_t Label)

BAL Salto sin condicion, de cualquier forma

Parámetros

<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.3. void BCC (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BCC salto si es menor (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.4. void BCS (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BCS Salto si es mayor o igual (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.5. void BEQ (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BEQ Salto si es igual

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.6. void BGE (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BGE Salto si es mayor o igual (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.7. void BGT (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BGT Salto si es mayor (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.8. void BHI (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BHI Salto si es mayor (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.9. void BL (uint32_t * *Registro*, uint32_t *Label*)

BL Salto guardando posicion instruccion siguiente en LR

Parámetros

<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.10. void BLE (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BLE Salto si es menor o igual (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.11. void BLS (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BLS Salto si es menor o igual (sin signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.12. void BLT (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BLT Salto si es menor (con signo)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.13. void BMI (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BMI Salto si es negativo

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.14. void BNE (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BNE Salto si no es igual

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.15. void BPL (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BPL Salto si es positivo

/fn void BPL(char **R_Banderas*,uint32_t **Registro*,uint32_t *Label*)

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.16. void BVC (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BVC Salto si no hay sobreflujo

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.17. void BVS (char * *R_Banderas*, uint32_t * *Registro*, uint32_t *Label*)

BVS Salto si hay sobreflujo

Parámetros

<i>*R_Banderas</i>	Puntero para el registro de las banderas
<i>*Registro</i>	Puntero a el registro
<i>Label</i>	Posicion de salto

5.4.1.18. void BX (uint32_t * *Registro*)

BX Salta a la posicion guardada en LR

Parámetros

<i>*Registro</i>	Puntero a el registro
------------------	-----------------------

5.5. Referencia del Archivo colors.h

'defines'

- #define BLACK 0 /* Black */
- #define BLUE 1 /* Blue */
- #define GREEN 2 /* Green */
- #define AQUA 3 /* Aqua */
- #define RED 4 /* Red */
- #define PURPLE 5 /* Purple */
- #define YELLOW 6 /* Yellow */
- #define WHITE 7 /* White */
- #define GRAY 8 /* Gray */
- #define LIGHT_BLUE 9 /* Light Blue */
- #define LIGHT_GREEN A /* Light Green */
- #define LIGHT_AQUA B /* Light Aqua */
- #define LIGHT_RED C /* Light Red */
- #define LIGHT_PURPLE D /* Light Purple */
- #define LIGHT_YELLOW E /* Light Yellow */
- #define BRIGHT_WHITE F /* Bright White */

5.5.1. Documentación de los 'defines'

5.5.1.1. #define AQUA 3 /* Aqua */

5.5.1.2. #define BLACK 0 /* Black */

5.5.1.3. #define BLUE 1 /* Blue */

5.5.1.4. #define BRIGHT_WHITE F /* Bright White */

5.5.1.5. #define GRAY 8 /* Gray */

5.5.1.6. #define GREEN 2 /* Green */

5.5.1.7. #define LIGHT_AQUA B /* Light Aqua */

5.5.1.8. #define LIGHT_BLUE 9 /* Light Blue */

5.5.1.9. `#define LIGHT_GREEN A /* Light Green */`

5.5.1.10. `#define LIGHT_PURPLE D /* Light Purple */`

5.5.1.11. `#define LIGHT_RED C /* Light Red */`

5.5.1.12. `#define LIGHT_YELLOW E /* Light Yellow */`

5.5.1.13. `#define PURPLE 5 /* Purple */`

5.5.1.14. `#define RED 4 /* Red */`

5.5.1.15. `#define WHITE 7 /* White */`

5.5.1.16. `#define YELLOW 6 /* Yellow */`

5.6. Referencia del Archivo curses.h

```
#include <stdarg.h>
#include <stddef.h>
#include <stdio.h>
```

Estructuras de datos

- struct [MOUSE_STATUS](#)
- struct [MEVENT](#)
- struct [_win](#)
- struct [SCREEN](#)

'defines'

- `#define PDC_BUILD 3401`
- `#define PDCURSES 1 /* PDCurses-only routines */`
- `#define XOPEN 1 /* X/Open Curses routines */`
- `#define SYSVcurses 1 /* System V Curses routines */`
- `#define BSDcurses 1 /* BSD Curses routines */`
- `#define CHTYPE_LONG 1 /* size of chtype; long */`
- `#define FALSE 0`
- `#define TRUE 1`
- `#define NULL (void *)0`
- `#define ERR (-1)`
- `#define OK 0`
- `#define BUTTON_RELEASED 0x0000`
- `#define BUTTON_PRESSED 0x0001`
- `#define BUTTON_CLICKED 0x0002`
- `#define BUTTON_DOUBLE_CLICKED 0x0003`
- `#define BUTTON_TRIPLE_CLICKED 0x0004`
- `#define BUTTON_MOVED 0x0005 /* PDCurses */`
- `#define WHEEL_SCROLLED 0x0006 /* PDCurses */`
- `#define BUTTON_ACTION_MASK 0x0007 /* PDCurses */`
- `#define PDC_BUTTON_SHIFT 0x0008 /* PDCurses */`
- `#define PDC_BUTTON_CONTROL 0x0010 /* PDCurses */`
- `#define PDC_BUTTON_ALT 0x0020 /* PDCurses */`

- #define `BUTTON_MODIFIER_MASK` 0x0038 /* PDCurses */
- #define `MOUSE_X_POS` (Mouse_status.x)
- #define `MOUSE_Y_POS` (Mouse_status.y)
- #define `PDC_MOUSE_MOVED` 0x0008
- #define `PDC_MOUSE_POSITION` 0x0010
- #define `PDC_MOUSE_WHEEL_UP` 0x0020
- #define `PDC_MOUSE_WHEEL_DOWN` 0x0040
- #define `A_BUTTON_CHANGED` (Mouse_status.changes & 7)
- #define `MOUSE_MOVED` (Mouse_status.changes & `PDC_MOUSE_MOVED`)
- #define `MOUSE_POS_REPORT` (Mouse_status.changes & `PDC_MOUSE_POSITION`)
- #define `BUTTON_CHANGED(x)` (Mouse_status.changes & (1 << ((x) - 1)))
- #define `BUTTON_STATUS(x)` (Mouse_status.button[(x) - 1])
- #define `MOUSE_WHEEL_UP` (Mouse_status.changes & `PDC_MOUSE_WHEEL_UP`)
- #define `MOUSE_WHEEL_DOWN` (Mouse_status.changes & `PDC_MOUSE_WHEEL_DOWN`)
- #define `BUTTON1_RELEASED` 0x00000001L
- #define `BUTTON1_PRESSED` 0x00000002L
- #define `BUTTON1_CLICKED` 0x00000004L
- #define `BUTTON1_DOUBLE_CLICKED` 0x00000008L
- #define `BUTTON1_TRIPLE_CLICKED` 0x00000010L
- #define `BUTTON1_MOVED` 0x00000010L /* PDCurses */
- #define `BUTTON2_RELEASED` 0x00000020L
- #define `BUTTON2_PRESSED` 0x00000040L
- #define `BUTTON2_CLICKED` 0x00000080L
- #define `BUTTON2_DOUBLE_CLICKED` 0x00000100L
- #define `BUTTON2_TRIPLE_CLICKED` 0x00000200L
- #define `BUTTON2_MOVED` 0x00000200L /* PDCurses */
- #define `BUTTON3_RELEASED` 0x00000400L
- #define `BUTTON3_PRESSED` 0x00000800L
- #define `BUTTON3_CLICKED` 0x00001000L
- #define `BUTTON3_DOUBLE_CLICKED` 0x00002000L
- #define `BUTTON3_TRIPLE_CLICKED` 0x00004000L
- #define `BUTTON3_MOVED` 0x00004000L /* PDCurses */
- #define `BUTTON4_RELEASED` 0x00008000L
- #define `BUTTON4_PRESSED` 0x00010000L
- #define `BUTTON4_CLICKED` 0x00020000L
- #define `BUTTON4_DOUBLE_CLICKED` 0x00040000L
- #define `BUTTON4_TRIPLE_CLICKED` 0x00080000L
- #define `BUTTON5_RELEASED` 0x00100000L
- #define `BUTTON5_PRESSED` 0x00200000L
- #define `BUTTON5_CLICKED` 0x00400000L
- #define `BUTTON5_DOUBLE_CLICKED` 0x00800000L
- #define `BUTTON5_TRIPLE_CLICKED` 0x01000000L
- #define `MOUSE_WHEEL_SCROLL` 0x02000000L /* PDCurses */
- #define `BUTTON_MODIFIER_SHIFT` 0x04000000L /* PDCurses */
- #define `BUTTON_MODIFIER_CONTROL` 0x08000000L /* PDCurses */
- #define `BUTTON_MODIFIER_ALT` 0x10000000L /* PDCurses */
- #define `ALL_MOUSE_EVENTS` 0x1fffffL
- #define `REPORT_MOUSE_POSITION` 0x20000000L
- #define `BUTTON_SHIFT` `PDC_BUTTON_SHIFT`
- #define `BUTTON_CONTROL` `PDC_BUTTON_CONTROL`
- #define `BUTTON_ALT` `PDC_BUTTON_ALT`
- #define `PDCEX` extern
- #define `A_NORMAL` (chtype)0
- #define `A_ALTCHARSET` (chtype)0x00010000
- #define `A_RIGHTLINE` (chtype)0x00020000

- #define `A_LEFTLINE` (`chtype`)0x00040000
- #define `A_INVIS` (`chtype`)0x00080000
- #define `A_UNDERLINE` (`chtype`)0x00100000
- #define `A_REVERSE` (`chtype`)0x00200000
- #define `A_BLINK` (`chtype`)0x00400000
- #define `A_BOLD` (`chtype`)0x00800000
- #define `A_ATTRIBUTES` (`chtype`)0xffff0000
- #define `A_CHARTEXT` (`chtype`)0x0000ffff
- #define `A_COLOR` (`chtype`)0xff000000
- #define `A_ITALIC` `A_INVIS`
- #define `A_PROTECT` (`A_UNDERLINE` | `A_LEFTLINE` | `A_RIGHTLINE`)
- #define `PDC_ATTR_SHIFT` 19
- #define `PDC_COLOR_SHIFT` 24
- #define `A_STANDOUT` (`A_REVERSE` | `A_BOLD`) /* X/Open */
- #define `A_DIM` `A_NORMAL`
- #define `CHR_MSK` `A_CHARTEXT` /* Obsolete */
- #define `ATR_MSK` `A_ATTRIBUTES` /* Obsolete */
- #define `ATR_NRM` `A_NORMAL` /* Obsolete */
- #define `WA_ALTCHARSET` `A_ALTCHARSET`
- #define `WA_BLINK` `A_BLINK`
- #define `WA_BOLD` `A_BOLD`
- #define `WA_DIM` `A_DIM`
- #define `WA_INVIS` `A_INVIS`
- #define `WA_LEFT` `A_LEFTLINE`
- #define `WA_PROTECT` `A_PROTECT`
- #define `WA_REVERSE` `A_REVERSE`
- #define `WA_RIGHT` `A_RIGHTLINE`
- #define `WA_STANDOUT` `A_STANDOUT`
- #define `WA_UNDERLINE` `A_UNDERLINE`
- #define `WA_HORIZONTAL` `A_NORMAL`
- #define `WA_LOW` `A_NORMAL`
- #define `WA_TOP` `A_NORMAL`
- #define `WA_VERTICAL` `A_NORMAL`
- #define `ACS_PICK`(`w`, `n`) ((`chtype`)`w` | `A_ALTCHARSET`)
- #define `ACS_ULCORNER` `ACS_PICK`('l', '+')
- #define `ACS_LLCORNER` `ACS_PICK`('m', '+')
- #define `ACS_URCORNER` `ACS_PICK`('k', '+')
- #define `ACS_LRCORNER` `ACS_PICK`('j', '+')
- #define `ACS_RTEE` `ACS_PICK`('u', '+')
- #define `ACS_LTEE` `ACS_PICK`('t', '+')
- #define `ACS_BTEE` `ACS_PICK`('v', '+')
- #define `ACS_TTEE` `ACS_PICK`('w', '+')
- #define `ACS_HLINE` `ACS_PICK`('q', '-')
- #define `ACS_VLINE` `ACS_PICK`('x', '|')
- #define `ACS_PLUS` `ACS_PICK`('n', '+')
- #define `ACS_S1` `ACS_PICK`('o', '-')
- #define `ACS_S9` `ACS_PICK`('s', '_')
- #define `ACS_DIAMOND` `ACS_PICK`('"', '+')
- #define `ACS_CKBOARD` `ACS_PICK`('a', ':')
- #define `ACS_DEGREE` `ACS_PICK`('f', '"')
- #define `ACS_PLMINUS` `ACS_PICK`('g', '#')
- #define `ACS_BULLET` `ACS_PICK`('~', 'o')
- #define `ACS_LARROW` `ACS_PICK`('<', '<')
- #define `ACS_RARROW` `ACS_PICK`('+', '>')
- #define `ACS_DARROW` `ACS_PICK`('.', 'v')

- #define ACS_UARROW ACS_PICK('-', '^')
- #define ACS_BOARD ACS_PICK('h', '#')
- #define ACS_LANTERN ACS_PICK('i', '*')
- #define ACS_BLOCK ACS_PICK('O', '#')
- #define ACS_S3 ACS_PICK('p', '-')
- #define ACS_S7 ACS_PICK('r', '-')
- #define ACS_LEQUAL ACS_PICK('y', '<')
- #define ACS_GEQUAL ACS_PICK('z', '>')
- #define ACS_PI ACS_PICK('{', 'n')
- #define ACS_NEQUAL ACS_PICK('|', '+')
- #define ACS_STERLING ACS_PICK('}', 'L')
- #define ACS_BSSB ACS_ULCORNER
- #define ACS_SSBB ACS_LLCORNER
- #define ACS_BBSS ACS_URCORNER
- #define ACS_SBBS ACS_LRCORNER
- #define ACS_SBSS ACS_RTEE
- #define ACS_SSSB ACS_LTEE
- #define ACS_SSBS ACS_BTEE
- #define ACS_BSSS ACS_TTEE
- #define ACS_BSBS ACS_HLINE
- #define ACS_SBSB ACS_VLINE
- #define ACS_SSSS ACS_PLUS
- #define COLOR_BLACK 0
- #define COLOR_BLUE 1
- #define COLOR_GREEN 2
- #define COLOR_RED 4
- #define COLOR_CYAN (COLOR_BLUE | COLOR_GREEN)
- #define COLOR_MAGENTA (COLOR_RED | COLOR_BLUE)
- #define COLOR_YELLOW (COLOR_RED | COLOR_GREEN)
- #define COLOR_WHITE 7
- #define KEY_CODE_YES 0x100 /* If get_wch() gives a key code */
- #define KEY_BREAK 0x101 /* Not on PC KBD */
- #define KEY_DOWN 0x102 /* Down arrow key */
- #define KEY_UP 0x103 /* Up arrow key */
- #define KEY_LEFT 0x104 /* Left arrow key */
- #define KEY_RIGHT 0x105 /* Right arrow key */
- #define KEY_HOME 0x106 /* home key */
- #define KEY_BACKSPACE 0x107 /* not on pc */
- #define KEY_F0 0x108 /* function keys; 64 reserved */
- #define KEY_DL 0x148 /* delete line */
- #define KEY_IL 0x149 /* insert line */
- #define KEY_DC 0x14a /* delete character */
- #define KEY_IC 0x14b /* insert char or enter ins mode */
- #define KEY_EIC 0x14c /* exit insert char mode */
- #define KEY_CLEAR 0x14d /* clear screen */
- #define KEY_EOS 0x14e /* clear to end of screen */
- #define KEY_EOL 0x14f /* clear to end of line */
- #define KEY_SF 0x150 /* scroll 1 line forward */
- #define KEY_SR 0x151 /* scroll 1 line back (reverse) */
- #define KEY_NPAGE 0x152 /* next page */
- #define KEY_PPAGE 0x153 /* previous page */
- #define KEY_STAB 0x154 /* set tab */
- #define KEY_CTAB 0x155 /* clear tab */
- #define KEY_CATAB 0x156 /* clear all tabs */
- #define KEY_ENTER 0x157 /* enter or send (unreliable) */

- #define `KEY_SRESET` 0x158 /* soft/reset (partial/unreliable) */
- #define `KEY_RESET` 0x159 /* reset/hard reset (unreliable) */
- #define `KEY_PRINT` 0x15a /* print/copy */
- #define `KEY_LL` 0x15b /* home down/bottom (lower left) */
- #define `KEY_ABORT` 0x15c /* abort/terminate key (any) */
- #define `KEY_SHELP` 0x15d /* short help */
- #define `KEY_LHELP` 0x15e /* long help */
- #define `KEY_BTAB` 0x15f /* Back tab key */
- #define `KEY_BEG` 0x160 /* beg(inning) key */
- #define `KEY_CANCEL` 0x161 /* cancel key */
- #define `KEY_CLOSE` 0x162 /* close key */
- #define `KEY_COMMAND` 0x163 /* cmd (command) key */
- #define `KEY_COPY` 0x164 /* copy key */
- #define `KEY_CREATE` 0x165 /* create key */
- #define `KEY_END` 0x166 /* end key */
- #define `KEY_EXIT` 0x167 /* exit key */
- #define `KEY_FIND` 0x168 /* find key */
- #define `KEY_HELP` 0x169 /* help key */
- #define `KEY_MARK` 0x16a /* mark key */
- #define `KEY_MESSAGE` 0x16b /* message key */
- #define `KEY_MOVE` 0x16c /* move key */
- #define `KEY_NEXT` 0x16d /* next object key */
- #define `KEY_OPEN` 0x16e /* open key */
- #define `KEY_OPTIONS` 0x16f /* options key */
- #define `KEY_PREVIOUS` 0x170 /* previous object key */
- #define `KEY_REDO` 0x171 /* redo key */
- #define `KEY_REFERENCE` 0x172 /* ref(ERENCE) key */
- #define `KEY_REFRESH` 0x173 /* refresh key */
- #define `KEY_REPLACE` 0x174 /* replace key */
- #define `KEY_RESTART` 0x175 /* restart key */
- #define `KEY_RESUME` 0x176 /* resume key */
- #define `KEY_SAVE` 0x177 /* save key */
- #define `KEY_SBEG` 0x178 /* shifted beginning key */
- #define `KEY_SCANCEL` 0x179 /* shifted cancel key */
- #define `KEY_SCOMMAND` 0x17a /* shifted command key */
- #define `KEY_SCOPY` 0x17b /* shifted copy key */
- #define `KEY_SCREATE` 0x17c /* shifted create key */
- #define `KEY_SDC` 0x17d /* shifted delete char key */
- #define `KEY_SDL` 0x17e /* shifted delete line key */
- #define `KEY_SELECT` 0x17f /* select key */
- #define `KEY_SEND` 0x180 /* shifted end key */
- #define `KEY_SEOL` 0x181 /* shifted clear line key */
- #define `KEY_SEXIT` 0x182 /* shifted exit key */
- #define `KEY_SFIND` 0x183 /* shifted find key */
- #define `KEY_SHOME` 0x184 /* shifted home key */
- #define `KEY_SIC` 0x185 /* shifted input key */
- #define `KEY_SLEFT` 0x187 /* shifted left arrow key */
- #define `KEY_SMESSAGE` 0x188 /* shifted message key */
- #define `KEY_SMOVE` 0x189 /* shifted move key */
- #define `KEY_SNEXT` 0x18a /* shifted next key */
- #define `KEY_SOPTIONS` 0x18b /* shifted options key */
- #define `KEY_SPREVIOUS` 0x18c /* shifted prev key */
- #define `KEY_SPRINT` 0x18d /* shifted print key */
- #define `KEY_SREDO` 0x18e /* shifted redo key */
- #define `KEY_SREPLACE` 0x18f /* shifted replace key */

- #define KEY_SRIGHT 0x190 /* shifted right arrow */
- #define KEY_SRSUME 0x191 /* shifted resume key */
- #define KEY_SSAVE 0x192 /* shifted save key */
- #define KEY_SSUSPEND 0x193 /* shifted suspend key */
- #define KEY_SUNDO 0x194 /* shifted undo key */
- #define KEY_SUSPEND 0x195 /* suspend key */
- #define KEY_UNDO 0x196 /* undo key */
- #define ALT_0 0x197
- #define ALT_1 0x198
- #define ALT_2 0x199
- #define ALT_3 0x19a
- #define ALT_4 0x19b
- #define ALT_5 0x19c
- #define ALT_6 0x19d
- #define ALT_7 0x19e
- #define ALT_8 0x19f
- #define ALT_9 0x1a0
- #define ALT_A 0x1a1
- #define ALT_B 0x1a2
- #define ALT_C 0x1a3
- #define ALT_D 0x1a4
- #define ALT_E 0x1a5
- #define ALT_F 0x1a6
- #define ALT_G 0x1a7
- #define ALT_H 0x1a8
- #define ALT_I 0x1a9
- #define ALT_J 0x1aa
- #define ALT_K 0x1ab
- #define ALT_L 0x1ac
- #define ALT_M 0x1ad
- #define ALT_N 0x1ae
- #define ALT_O 0x1af
- #define ALT_P 0x1b0
- #define ALT_Q 0x1b1
- #define ALT_R 0x1b2
- #define ALT_S 0x1b3
- #define ALT_T 0x1b4
- #define ALT_U 0x1b5
- #define ALT_V 0x1b6
- #define ALT_W 0x1b7
- #define ALT_X 0x1b8
- #define ALT_Y 0x1b9
- #define ALT_Z 0x1ba
- #define CTL_LEFT 0x1bb /* Control-Left-Arrow */
- #define CTL_RIGHT 0x1bc
- #define CTL_PGUP 0x1bd
- #define CTL_PGDN 0x1be
- #define CTL_HOME 0x1bf
- #define CTL_END 0x1c0
- #define KEY_A1 0x1c1 /* upper left on Virtual keypad */
- #define KEY_A2 0x1c2 /* upper middle on Virt. keypad */
- #define KEY_A3 0x1c3 /* upper right on Vir. keypad */
- #define KEY_B1 0x1c4 /* middle left on Virt. keypad */
- #define KEY_B2 0x1c5 /* center on Virt. keypad */
- #define KEY_B3 0x1c6 /* middle right on Vir. keypad */

- #define KEY_C1 0x1c7 /* lower left on Virt. keypad */
- #define KEY_C2 0x1c8 /* lower middle on Virt. keypad */
- #define KEY_C3 0x1c9 /* lower right on Vir. keypad */
- #define PADSLASH 0x1ca /* slash on keypad */
- #define PADENTER 0x1cb /* enter on keypad */
- #define CTL_PADENTER 0x1cc /* ctl-enter on keypad */
- #define ALT_PADENTER 0x1cd /* alt-enter on keypad */
- #define PADSTOP 0x1ce /* stop on keypad */
- #define PADSTAR 0x1cf /* star on keypad */
- #define PADMINUS 0x1d0 /* minus on keypad */
- #define PADPLUS 0x1d1 /* plus on keypad */
- #define CTL_PADSTOP 0x1d2 /* ctl-stop on keypad */
- #define CTL_PADCENTER 0x1d3 /* ctl-enter on keypad */
- #define CTL_PADPLUS 0x1d4 /* ctl-plus on keypad */
- #define CTL_PADMINUS 0x1d5 /* ctl-minus on keypad */
- #define CTL_PADSLASH 0x1d6 /* ctl-slash on keypad */
- #define CTL_PADSTAR 0x1d7 /* ctl-star on keypad */
- #define ALT_PADPLUS 0x1d8 /* alt-plus on keypad */
- #define ALT_PADMINUS 0x1d9 /* alt-minus on keypad */
- #define ALT_PADSLASH 0x1da /* alt-slash on keypad */
- #define ALT_PADSTAR 0x1db /* alt-star on keypad */
- #define ALT_PADSTOP 0x1dc /* alt-stop on keypad */
- #define CTL_INS 0x1dd /* ctl-insert */
- #define ALT_DEL 0x1de /* alt-delete */
- #define ALT_INS 0x1df /* alt-insert */
- #define CTL_UP 0x1e0 /* ctl-up arrow */
- #define CTL_DOWN 0x1e1 /* ctl-down arrow */
- #define CTL_TAB 0x1e2 /* ctl-tab */
- #define ALT_TAB 0x1e3
- #define ALT_MINUS 0x1e4
- #define ALT_EQUAL 0x1e5
- #define ALT_HOME 0x1e6
- #define ALT_PGUP 0x1e7
- #define ALT_PGDN 0x1e8
- #define ALT_END 0x1e9
- #define ALT_UP 0x1ea /* alt-up arrow */
- #define ALT_DOWN 0x1eb /* alt-down arrow */
- #define ALT_RIGHT 0x1ec /* alt-right arrow */
- #define ALT_LEFT 0x1ed /* alt-left arrow */
- #define ALT_ENTER 0x1ee /* alt-enter */
- #define ALT_ESC 0x1ef /* alt-escape */
- #define ALT_BQUOTE 0x1f0 /* alt-back quote */
- #define ALT_LBRACKET 0x1f1 /* alt-left bracket */
- #define ALT_RBRACKET 0x1f2 /* alt-right bracket */
- #define ALT_SEMICOLON 0x1f3 /* alt-semi-colon */
- #define ALT_FQUOTE 0x1f4 /* alt-forward quote */
- #define ALT_COMMA 0x1f5 /* alt-comma */
- #define ALT_STOP 0x1f6 /* alt-stop */
- #define ALT_FSLASH 0x1f7 /* alt-forward slash */
- #define ALT_BKSP 0x1f8 /* alt-backspace */
- #define CTL_BKSP 0x1f9 /* ctl-backspace */
- #define PAD0 0x1fa /* keypad 0 */
- #define CTL_PAD0 0x1fb /* ctl-keypad 0 */
- #define CTL_PAD1 0x1fc
- #define CTL_PAD2 0x1fd

- #define CTL_PAD3 0x1fe
- #define CTL_PAD4 0x1ff
- #define CTL_PAD5 0x200
- #define CTL_PAD6 0x201
- #define CTL_PAD7 0x202
- #define CTL_PAD8 0x203
- #define CTL_PAD9 0x204
- #define ALT_PAD0 0x205 /* alt-keypad 0 */
- #define ALT_PAD1 0x206
- #define ALT_PAD2 0x207
- #define ALT_PAD3 0x208
- #define ALT_PAD4 0x209
- #define ALT_PAD5 0x20a
- #define ALT_PAD6 0x20b
- #define ALT_PAD7 0x20c
- #define ALT_PAD8 0x20d
- #define ALT_PAD9 0x20e
- #define CTL_DEL 0x20f /* clt-delete */
- #define ALT_BSLASH 0x210 /* alt-back slash */
- #define CTL_ENTER 0x211 /* ctl-enter */
- #define SHF_PADENTER 0x212 /* shift-enter on keypad */
- #define SHF_PADSLASH 0x213 /* shift-slash on keypad */
- #define SHF_PADSTAR 0x214 /* shift-star on keypad */
- #define SHF_PADPLUS 0x215 /* shift-plus on keypad */
- #define SHF_PADMINUS 0x216 /* shift-minus on keypad */
- #define SHF_UP 0x217 /* shift-up on keypad */
- #define SHF_DOWN 0x218 /* shift-down on keypad */
- #define SHF_IC 0x219 /* shift-insert on keypad */
- #define SHF_DC 0x21a /* shift-delete on keypad */
- #define KEY_MOUSE 0x21b /* "mouse" key */
- #define KEY_SHIFT_L 0x21c /* Left-shift */
- #define KEY_SHIFT_R 0x21d /* Right-shift */
- #define KEY_CONTROL_L 0x21e /* Left-control */
- #define KEY_CONTROL_R 0x21f /* Right-control */
- #define KEY_ALT_L 0x220 /* Left-alt */
- #define KEY_ALT_R 0x221 /* Right-alt */
- #define KEY_RESIZE 0x222 /* Window resize */
- #define KEY_SUP 0x223 /* Shifted up arrow */
- #define KEY_SDOWN 0x224 /* Shifted down arrow */
- #define KEY_MIN KEY_BREAK /* Minimum curses key value */
- #define KEY_MAX KEY_SDOWN /* Maximum curses key */
- #define KEY_F(n) (KEY_F0 + (n))
- #define getch() wgetch(stdscr)
- #define ungetch(ch) PDC_ungetch(ch)
- #define COLOR_PAIR(n) (((chtype)(n) << PDC_COLOR_SHIFT) & A_COLOR)
- #define PAIR_NUMBER(n) (((n) & A_COLOR) >> PDC_COLOR_SHIFT)
- #define getbegyx(w, y, x) (y = getbegy(w), x = getbegx(w))
- #define getmaxyx(w, y, x) (y = getmaxy(w), x = getmaxx(w))
- #define getparyx(w, y, x) (y = getpary(w), x = getparx(w))
- #define getyx(w, y, x) (y = getcury(w), x = getcurx(w))
- #define getsyx(y, x)
- #define PDC_CLIP_SUCCESS 0
- #define PDC_CLIP_ACCESS_ERROR 1
- #define PDC_CLIP_EMPTY 2
- #define PDC_CLIP_MEMORY_ERROR 3

- #define PDC_KEY_MODIFIER_SHIFT 1
- #define PDC_KEY_MODIFIER_CONTROL 2
- #define PDC_KEY_MODIFIER_ALT 4
- #define PDC_KEY_MODIFIER_NUMLOCK 8

'typedefs'

- typedef unsigned char bool
- typedef unsigned long chtype
- typedef chtype attr_t
- typedef unsigned long mmask_t
- typedef struct _win WINDOW

Funciones

- int addch (const chtype)
- int addchnstr (const chtype *, int)
- int addchstr (const chtype *)
- int addnstr (const char *, int)
- int addstr (const char *)
- int attroff (chtype)
- int attron (chtype)
- int attrset (chtype)
- int attr_get (attr_t *, short *, void *)
- int attr_off (attr_t, void *)
- int attr_on (attr_t, void *)
- int attr_set (attr_t, short, void *)
- int baudrate (void)
- int beep (void)
- int bkgd (chtype)
- void bkgdset (chtype)
- int border (chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype)
- int box (WINDOW *, chtype, chtype)
- bool can_change_color (void)
- int cbreak (void)
- int chgat (int, attr_t, short, const void *)
- int clearok (WINDOW *, bool)
- int clear (void)
- int clrtoebot (void)
- int clrtoeol (void)
- int color_content (short, short *, short *, short *)
- int color_set (short, void *)
- int copywin (const WINDOW *, WINDOW *, int, int, int, int, int, int)
- int curs_set (int)
- int def_prog_mode (void)
- int def_shell_mode (void)
- int delay_output (int)
- int delch (void)
- int deleteln (void)
- void delscreen (SCREEN *)
- int delwin (WINDOW *)
- WINDOW * derwin (WINDOW *, int, int, int, int)
- int doupdate (void)
- WINDOW * dupwin (WINDOW *)

- int `echochar` (const `chtype`)
- int `echo` (void)
- int `endwin` (void)
- char `erasechar` (void)
- int `erase` (void)
- void `filter` (void)
- int `flash` (void)
- int `flushinp` (void)
- `chtype` `getbkgd` (`WINDOW *`)
- int `getnstr` (char *, int)
- int `getstr` (char *)
- `WINDOW *` `getwin` (FILE *)
- int `halfdelay` (int)
- bool `has_colors` (void)
- bool `has_ic` (void)
- bool `has_il` (void)
- int `hline` (`chtype`, int)
- void `idcok` (`WINDOW *`, bool)
- int `idlok` (`WINDOW *`, bool)
- void `immedok` (`WINDOW *`, bool)
- int `inchnstr` (`chtype *`, int)
- int `inchstr` (`chtype *`)
- `chtype` `inch` (void)
- int `init_color` (short, short, short, short)
- int `init_pair` (short, short, short)
- `WINDOW *` `initscr` (void)
- int `innstr` (char *, int)
- int `insch` (`chtype`)
- int `insdelin` (int)
- int `insertln` (void)
- int `insnstr` (const char *, int)
- int `insstr` (const char *)
- int `instr` (char *)
- int `intrflush` (`WINDOW *`, bool)
- bool `isendwin` (void)
- bool `is_linetouched` (`WINDOW *`, int)
- bool `is_wintouched` (`WINDOW *`)
- char * `keyname` (int)
- int `keypad` (`WINDOW *`, bool)
- char `killchar` (void)
- int `leaveok` (`WINDOW *`, bool)
- char * `longname` (void)
- int `meta` (`WINDOW *`, bool)
- int `move` (int, int)
- int `mvaddch` (int, int, const `chtype`)
- int `mvaddchnstr` (int, int, const `chtype *`, int)
- int `mvaddchstr` (int, int, const `chtype *`)
- int `mvaddnstr` (int, int, const char *, int)
- int `mvaddstr` (int, int, const char *)
- int `mvchgat` (int, int, int, `attr_t`, short, const void *)
- int `mvcur` (int, int, int, int)
- int `mvdelch` (int, int)
- int `mvderwin` (`WINDOW *`, int, int)
- int `mvgetch` (int, int)
- int `mvgetnstr` (int, int, char *, int)

- int `mvgetstr` (int, int, char *)
- int `mvhline` (int, int, `chtype`, int)
- `chtype` `mvinch` (int, int)
- int `mvinchnstr` (int, int, `chtype` *, int)
- int `mvinchstr` (int, int, `chtype` *)
- int `mvinnstr` (int, int, char *, int)
- int `mvinsch` (int, int, `chtype`)
- int `mvinsnstr` (int, int, const char *, int)
- int `mvinsstr` (int, int, const char *)
- int `mvinstr` (int, int, char *)
- int `mvprintw` (int, int, const char *,...)
- int `mvscanw` (int, int, const char *,...)
- int `mvvline` (int, int, `chtype`, int)
- int `mwaddchnstr` (`WINDOW` *, int, int, const `chtype` *, int)
- int `mwaddchstr` (`WINDOW` *, int, int, const `chtype` *)
- int `mwaddch` (`WINDOW` *, int, int, const `chtype`)
- int `mwaddnstr` (`WINDOW` *, int, int, const char *, int)
- int `mwaddstr` (`WINDOW` *, int, int, const char *)
- int `mwchgat` (`WINDOW` *, int, int, int, `attr_t`, short, const void *)
- int `mwdelch` (`WINDOW` *, int, int)
- int `mwgetch` (`WINDOW` *, int, int)
- int `mwgetnstr` (`WINDOW` *, int, int, char *, int)
- int `mwgetstr` (`WINDOW` *, int, int, char *)
- int `mwwhline` (`WINDOW` *, int, int, `chtype`, int)
- int `mwwinchnstr` (`WINDOW` *, int, int, `chtype` *, int)
- int `mwwinchstr` (`WINDOW` *, int, int, `chtype` *)
- `chtype` `mwwinch` (`WINDOW` *, int, int)
- int `mwwinstr` (`WINDOW` *, int, int, char *, int)
- int `mwwinsch` (`WINDOW` *, int, int, `chtype`)
- int `mwinsnstr` (`WINDOW` *, int, int, const char *, int)
- int `mwinsstr` (`WINDOW` *, int, int, const char *)
- int `mwinstr` (`WINDOW` *, int, int, char *)
- int `mwwin` (`WINDOW` *, int, int)
- int `mwprintw` (`WINDOW` *, int, int, const char *,...)
- int `mwscanw` (`WINDOW` *, int, int, const char *,...)
- int `mwvline` (`WINDOW` *, int, int, `chtype`, int)
- int `napms` (int)
- `WINDOW` * `newpad` (int, int)
- `SCREEN` * `newterm` (const char *, FILE *, FILE *)
- `WINDOW` * `newwin` (int, int, int, int)
- int `nl` (void)
- int `nocbreak` (void)
- int `nodelay` (`WINDOW` *, `bool`)
- int `noecho` (void)
- int `nonl` (void)
- void `noqiflush` (void)
- int `noraw` (void)
- int `notimeout` (`WINDOW` *, `bool`)
- int `overlay` (const `WINDOW` *, `WINDOW` *)
- int `overwrite` (const `WINDOW` *, `WINDOW` *)
- int `pair_content` (short, short *, short *)
- int `pechochar` (`WINDOW` *, `chtype`)
- int `pnoutrefresh` (`WINDOW` *, int, int, int, int, int, int)
- int `prefresh` (`WINDOW` *, int, int, int, int, int, int)
- int `printw` (const char *,...)

- int `putwin` (`WINDOW *`, `FILE *`)
- void `qiflush` (`void`)
- int `raw` (`void`)
- int `redrawwin` (`WINDOW *`)
- int `refresh` (`void`)
- int `reset_prog_mode` (`void`)
- int `reset_shell_mode` (`void`)
- int `resetty` (`void`)
- int `riporffline` (`int`, `int(*)`)(`WINDOW *`, `int`)
- int `savetty` (`void`)
- int `scanw` (`const char *`,...)
- int `scr_dump` (`const char *`)
- int `scr_init` (`const char *`)
- int `scr_restore` (`const char *`)
- int `scr_set` (`const char *`)
- int `scr1` (`int`)
- int `scroll` (`WINDOW *`)
- int `scrollok` (`WINDOW *`, `bool`)
- `SCREEN *` `set_term` (`SCREEN *`)
- int `setscrreg` (`int`, `int`)
- int `slk_attroff` (`const chtype`)
- int `slk_attr_off` (`const attr_t`, `void *`)
- int `slk_attron` (`const chtype`)
- int `slk_attr_on` (`const attr_t`, `void *`)
- int `slk_attrset` (`const chtype`)
- int `slk_attr_set` (`const attr_t`, `short`, `void *`)
- int `slk_clear` (`void`)
- int `slk_color` (`short`)
- int `slk_init` (`int`)
- `char *` `slk_label` (`int`)
- int `slk_noutrefresh` (`void`)
- int `slk_refresh` (`void`)
- int `slk_restore` (`void`)
- int `slk_set` (`int`, `const char *`, `int`)
- int `slk_touch` (`void`)
- int `standend` (`void`)
- int `standout` (`void`)
- int `start_color` (`void`)
- `WINDOW *` `subpad` (`WINDOW *`, `int`, `int`, `int`, `int`)
- `WINDOW *` `subwin` (`WINDOW *`, `int`, `int`, `int`, `int`)
- int `syncok` (`WINDOW *`, `bool`)
- `chtype` `termattrs` (`void`)
- `attr_t` `term_attrs` (`void`)
- `char *` `termname` (`void`)
- void `timeout` (`int`)
- int `touchline` (`WINDOW *`, `int`, `int`)
- int `touchwin` (`WINDOW *`)
- int `typeahead` (`int`)
- int `untouchwin` (`WINDOW *`)
- void `use_env` (`bool`)
- int `vidattr` (`chtype`)
- int `vid_attr` (`attr_t`, `short`, `void *`)
- int `vidputs` (`chtype`, `int(*)`)(`int`)
- int `vid_puts` (`attr_t`, `short`, `void *`, `int(*)`)(`int`)
- int `vline` (`chtype`, `int`)

- `int vw_printw (WINDOW *, const char *, va_list)`
- `int vwprintw (WINDOW *, const char *, va_list)`
- `int vw_scanw (WINDOW *, const char *, va_list)`
- `int vwscanw (WINDOW *, const char *, va_list)`
- `int waddchnstr (WINDOW *, const chtype *, int)`
- `int waddchstr (WINDOW *, const chtype *)`
- `int waddch (WINDOW *, const chtype)`
- `int waddnstr (WINDOW *, const char *, int)`
- `int waddstr (WINDOW *, const char *)`
- `int wattroff (WINDOW *, chtype)`
- `int wattron (WINDOW *, chtype)`
- `int wattrset (WINDOW *, chtype)`
- `int wattr_get (WINDOW *, attr_t *, short *, void *)`
- `int wattr_off (WINDOW *, attr_t, void *)`
- `int wattr_on (WINDOW *, attr_t, void *)`
- `int wattr_set (WINDOW *, attr_t, short, void *)`
- `void wbkgdset (WINDOW *, chtype)`
- `int wbkgd (WINDOW *, chtype)`
- `int wborder (WINDOW *, chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype)`
- `int wchgat (WINDOW *, int, attr_t, short, const void *)`
- `int wclear (WINDOW *)`
- `int wclrtobot (WINDOW *)`
- `int wclrtoeol (WINDOW *)`
- `int wcolor_set (WINDOW *, short, void *)`
- `void wcursyncup (WINDOW *)`
- `int wdelch (WINDOW *)`
- `int wdeleteln (WINDOW *)`
- `int wechochar (WINDOW *, const chtype)`
- `int werase (WINDOW *)`
- `int wgetch (WINDOW *)`
- `int wgetnstr (WINDOW *, char *, int)`
- `int wgetstr (WINDOW *, char *)`
- `int whline (WINDOW *, chtype, int)`
- `int winchnstr (WINDOW *, chtype *, int)`
- `int winchstr (WINDOW *, chtype *)`
- `chtype winch (WINDOW *)`
- `int winnstr (WINDOW *, char *, int)`
- `int winsch (WINDOW *, chtype)`
- `int winsdelln (WINDOW *, int)`
- `int winsertln (WINDOW *)`
- `int winsnstr (WINDOW *, const char *, int)`
- `int winsstr (WINDOW *, const char *)`
- `int winstr (WINDOW *, char *)`
- `int wmove (WINDOW *, int, int)`
- `int wnoutrefresh (WINDOW *)`
- `int wprintw (WINDOW *, const char *,...)`
- `int wredrawln (WINDOW *, int, int)`
- `int wrefresh (WINDOW *)`
- `int wscanw (WINDOW *, const char *,...)`
- `int wscr1 (WINDOW *, int)`
- `int wsetscrreg (WINDOW *, int, int)`
- `int wstandend (WINDOW *)`
- `int wstandout (WINDOW *)`
- `void wsyncdown (WINDOW *)`
- `void wsyncup (WINDOW *)`

- void `wtimeout` (`WINDOW *`, int)
- int `wtouchln` (`WINDOW *`, int, int, int)
- int `wvline` (`WINDOW *`, `chtype`, int)
- `chtype` `getattrs` (`WINDOW *`)
- int `getbegx` (`WINDOW *`)
- int `getbegy` (`WINDOW *`)
- int `getmaxx` (`WINDOW *`)
- int `getmaxy` (`WINDOW *`)
- int `getparx` (`WINDOW *`)
- int `getpary` (`WINDOW *`)
- int `getcurx` (`WINDOW *`)
- int `getcury` (`WINDOW *`)
- void `traceoff` (void)
- void `traceon` (void)
- char * `unctrl` (`chtype`)
- int `crmode` (void)
- int `nocrmode` (void)
- int `draino` (int)
- int `resetterm` (void)
- int `fixterm` (void)
- int `saveterm` (void)
- int `setsyx` (int, int)
- int `mouse_set` (unsigned long)
- int `mouse_on` (unsigned long)
- int `mouse_off` (unsigned long)
- int `request_mouse_pos` (void)
- int `map_button` (unsigned long)
- void `wmouse_position` (`WINDOW *`, int *, int *)
- unsigned long `getmouse` (void)
- unsigned long `getbmap` (void)
- int `assume_default_colors` (int, int)
- const char * `curses_version` (void)
- bool `has_key` (int)
- int `use_default_colors` (void)
- int `wresize` (`WINDOW *`, int, int)
- int `mouseinterval` (int)
- `mmask_t` `mousemask` (`mmask_t`, `mmask_t *`)
- bool `mouse_trafo` (int *, int *, bool)
- int `nc_getmouse` (`MEVENT *`)
- int `ungetmouse` (`MEVENT *`)
- bool `wenclose` (const `WINDOW *`, int, int)
- bool `wmouse_trafo` (const `WINDOW *`, int *, int *, bool)
- int `addrawch` (`chtype`)
- int `insrawch` (`chtype`)
- bool `is_termresized` (void)
- int `mvaddrawch` (int, int, `chtype`)
- int `mvdeleteln` (int, int)
- int `mvinsertln` (int, int)
- int `mvinsrawch` (int, int, `chtype`)
- int `mvwaddrawch` (`WINDOW *`, int, int, `chtype`)
- int `mvwdeleteln` (`WINDOW *`, int, int)
- int `mvwininsertln` (`WINDOW *`, int, int)
- int `mvwinsrawch` (`WINDOW *`, int, int, `chtype`)
- int `raw_output` (bool)
- int `resize_term` (int, int)

- `WINDOW * resize_window (WINDOW *, int, int)`
- `int waddrawch (WINDOW *, chtype)`
- `int winsrawch (WINDOW *, chtype)`
- `char wordchar (void)`
- `void PDC_debug (const char *,...)`
- `int PDC_ungetch (int)`
- `int PDC_set_blink (bool)`
- `int PDC_set_line_color (short)`
- `void PDC_set_title (const char *)`
- `int PDC_clearclipboard (void)`
- `int PDC_freeclipboard (char *)`
- `int PDC_getclipboard (char **, long *)`
- `int PDC_setclipboard (const char *, long)`
- `unsigned long PDC_get_input_fd (void)`
- `unsigned long PDC_get_key_modifiers (void)`
- `int PDC_return_key_modifiers (bool)`
- `int PDC_save_key_modifiers (bool)`

Variables

- `PDCEX int LINES`
- `PDCEX int COLS`
- `PDCEX WINDOW * stdscr`
- `PDCEX WINDOW * curscr`
- `PDCEX SCREEN * SP`
- `PDCEX MOUSE_STATUS Mouse_status`
- `PDCEX int COLORS`
- `PDCEX int COLOR_PAIRS`
- `PDCEX int TABSIZE`
- `PDCEX chtype acs_map []`
- `PDCEX char ttytype []`

5.6.1. Documentación de los 'defines'

- 5.6.1.1. `#define A_ALTCHARSET (chtype)0x00010000`
- 5.6.1.2. `#define A_ATTRIBUTES (chtype)0xffff0000`
- 5.6.1.3. `#define A_BLINK (chtype)0x00400000`
- 5.6.1.4. `#define A_BOLD (chtype)0x00800000`
- 5.6.1.5. `#define A_BUTTON_CHANGED (Mouse_status.changes & 7)`
- 5.6.1.6. `#define A_CHARTEXT (chtype)0x0000ffff`
- 5.6.1.7. `#define A_COLOR (chtype)0xff000000`
- 5.6.1.8. `#define A_DIM_NORMAL@`
- 5.6.1.9. `#define A_INVIS (chtype)0x00080000`
- 5.6.1.10. `#define A_ITALIC_INVIS@`

5.6.1.11. `#define A_LEFTLINE (chtype)0x00040000`

5.6.1.12. `#define A_NORMAL (chtype)0`

5.6.1.13. `#define A_PROTECT (A_UNDERLINE | A_LEFTLINE | A_RIGHTLINE)`

5.6.1.14. `#define A_REVERSE (chtype)0x00200000`

5.6.1.15. `#define A_RIGHTLINE (chtype)0x00020000`

5.6.1.16. `#define A_STANDOUT (A_REVERSE | A_BOLD) /* X/Open */`

5.6.1.17. `#define A_UNDERLINE (chtype)0x00100000`

5.6.1.18. `#define ACS_BBSCS_URCORNER@`

5.6.1.19. `#define ACS_BLOCKCS_PICK@('0', '#')`

5.6.1.20. `#define ACS_BOARDCS_PICK@('h', '#')`

5.6.1.21. `#define ACS_BSBSCS_HLINE@`

5.6.1.22. `#define ACS_BSSBCS_ULCORNER@`

5.6.1.23. `#define ACS_BSSSCS_TTEE@`

5.6.1.24. `#define ACS_BTEECS_PICK@('v', '+')`

5.6.1.25. `#define ACS_BULLETCES_PICK@('~', 'o')`

5.6.1.26. `#define ACS_CKBOARDCS_PICK@('a', ':')`

5.6.1.27. `#define ACS_DARROWCS_PICK@('.', 'v')`

5.6.1.28. `#define ACS_DEGREECS_PICK@('f', '\')`

5.6.1.29. `#define ACS_DIAMONDCS_PICK@('i', '+')`

5.6.1.30. `#define ACS_GEQUALCS_PICK@('z', '>')`

5.6.1.31. `#define ACS_HLINECS_PICK@('q', '-')`

5.6.1.32. `#define ACS_LANTERNCS_PICK@('l', '*')`

5.6.1.33. `#define ACS_LARROWCS_PICK@(',', '<')`

5.6.1.34. `#define ACS_LEQUALCS_PICK@('y', '<')`

5.6.1.35. `#define ACS_LLCORNERCS_PICK@('m', '+')`

5.6.1.36. `#define ACS_LRCORNERCS_PICK@('j', '+')`

5.6.1.37. `#define ACS_LTEECS_PICK@('t', '+')`

5.6.1.38. `#define ACS_NEQUALCS_PICK@('|', '+')`

5.6.1.39. `#define ACS_PICS_PICK@('{', 'n')`

5.6.1.40. `#define ACS_PICK(w, n) ((chtype)w | A_ALTCHARSET)`

5.6.1.41. `#define ACS_PLMINUSCS_PICK@('g', '#')`

5.6.1.42. `#define ACS_PLUSCS_PICK@('n', '+')`

5.6.1.43. `#define ACS_RARROWCS_PICK@('+', '>')`

5.6.1.44. `#define ACS_RTEECs_PICK@('u', '+')`

5.6.1.45. `#define ACS_S1CS_PICK@('o', '-')`

5.6.1.46. `#define ACS_S3CS_PICK@('p', '-')`

5.6.1.47. `#define ACS_S7CS_PICK@('r', '-')`

5.6.1.48. `#define ACS_S9CS_PICK@('s', '_')`

5.6.1.49. `#define ACS_SBBSCS_LRCORNER@`

5.6.1.50. `#define ACS_SBSBCS_VLINE@`

5.6.1.51. `#define ACS_SBSSCS_RTEE@`

5.6.1.52. `#define ACS_SBBBCS_LLCORNER@`

5.6.1.53. `#define ACS_SSBSCS_BTEE@`

5.6.1.54. `#define ACS_SSSBCS_LTEE@`

5.6.1.55. `#define ACS_SSSSCS_PLUS@`

5.6.1.56. `#define ACS_STERLINGCS_PICK@('}', 'L')`

5.6.1.57. `#define ACS_TTEECs_PICK@('w', '+')`

5.6.1.58. `#define ACS_UARROWCS_PICK@('-', '^')`

5.6.1.59. `#define ACS_ULCORNERCS_PICK@('l', '+')`

5.6.1.60. `#define ACS_URCORNERCS_PICK@('k', '+')`

5.6.1.61. `#define ACS_VLINECS_PICK@('x', '|')`

5.6.1.62. `#define ALL_MOUSE_EVENTS 0xffffffffL`

5.6.1.63. `#define ALT_0 0x197`

5.6.1.64. `#define ALT_1 0x198`

5.6.1.65. `#define ALT_2 0x199`

5.6.1.66. `#define ALT_3 0x19a`

5.6.1.67. `#define ALT_4 0x19b`

5.6.1.68. `#define ALT_5 0x19c`

5.6.1.69. `#define ALT_6 0x19d`

5.6.1.70. `#define ALT_7 0x19e`

5.6.1.71. `#define ALT_8 0x19f`

5.6.1.72. `#define ALT_9 0x1a0`

5.6.1.73. `#define ALT_A 0x1a1`

5.6.1.74. `#define ALT_B 0x1a2`

5.6.1.75. `#define ALT_BKSP 0x1f8 /* alt-backspace */`

5.6.1.76. `#define ALT_BQUOTE 0x1f0 /* alt-back quote */`

5.6.1.77. `#define ALT_BSLASH 0x210 /* alt-back slash */`

5.6.1.78. `#define ALT_C 0x1a3`

5.6.1.79. `#define ALT_COMMA 0x1f5 /* alt-comma */`

5.6.1.80. `#define ALT_D 0x1a4`

5.6.1.81. `#define ALT_DEL 0x1de /* alt-delete */`

5.6.1.82. `#define ALT_DOWN 0x1eb /* alt-down arrow */`

5.6.1.83. `#define ALT_E 0x1a5`

5.6.1.84. `#define ALT_END 0x1e9`

5.6.1.85. `#define ALT_ENTER 0x1ee /* alt-enter */`

5.6.1.86. `#define ALT_EQUAL 0x1e5`

5.6.1.87. `#define ALT_ESC 0x1ef /* alt-escape */`

5.6.1.88. `#define ALT_F 0x1a6`

5.6.1.89. `#define ALT_FQUOTE 0x1f4 /* alt-forward quote */`

5.6.1.90. `#define ALT_FSLASH 0x1f7 /* alt-forward slash */`

5.6.1.91. `#define ALT_G 0x1a7`

5.6.1.92. `#define ALT_H 0x1a8`

5.6.1.93. `#define ALT_HOME 0x1e6`

5.6.1.94. `#define ALT_I 0x1a9`

```
5.6.1.95. #define ALT_INS 0x1df /* alt-insert */

5.6.1.96. #define ALT_J 0x1aa

5.6.1.97. #define ALT_K 0x1ab

5.6.1.98. #define ALT_L 0x1ac

5.6.1.99. #define ALT_LBRACKET 0x1f1 /* alt-left bracket */

5.6.1.100. #define ALT_LEFT 0x1ed /* alt-left arrow */

5.6.1.101. #define ALT_M 0x1ad

5.6.1.102. #define ALT_MINUS 0x1e4

5.6.1.103. #define ALT_N 0x1ae

5.6.1.104. #define ALT_O 0x1af

5.6.1.105. #define ALT_P 0x1b0

5.6.1.106. #define ALT_PAD0 0x205 /* alt-keypad 0 */

5.6.1.107. #define ALT_PAD1 0x206

5.6.1.108. #define ALT_PAD2 0x207

5.6.1.109. #define ALT_PAD3 0x208

5.6.1.110. #define ALT_PAD4 0x209

5.6.1.111. #define ALT_PAD5 0x20a

5.6.1.112. #define ALT_PAD6 0x20b

5.6.1.113. #define ALT_PAD7 0x20c

5.6.1.114. #define ALT_PAD8 0x20d

5.6.1.115. #define ALT_PAD9 0x20e

5.6.1.116. #define ALT_PADENTER 0x1cd /* alt-enter on keypad */

5.6.1.117. #define ALT_PADMINUS 0x1d9 /* alt-minus on keypad */

5.6.1.118. #define ALT_PADPLUS 0x1d8 /* alt-plus on keypad */

5.6.1.119. #define ALT_PADSLASH 0x1da /* alt-slash on keypad */

5.6.1.120. #define ALT_PADSTAR 0x1db /* alt-star on keypad */

5.6.1.121. #define ALT_PADSTOP 0x1dc /* alt-stop on keypad */

5.6.1.122. #define ALT_PGDN 0x1e8
```

5.6.1.123. `#define ALT_PGUP 0x1e7`

5.6.1.124. `#define ALT_Q 0x1b1`

5.6.1.125. `#define ALT_R 0x1b2`

5.6.1.126. `#define ALT_RBRACKET 0x1f2 /* alt-right bracket */`

5.6.1.127. `#define ALT_RIGHT 0x1ec /* alt-right arrow */`

5.6.1.128. `#define ALT_S 0x1b3`

5.6.1.129. `#define ALT_SEMICOLON 0x1f3 /* alt-semi-colon */`

5.6.1.130. `#define ALT_STOP 0x1f6 /* alt-stop */`

5.6.1.131. `#define ALT_T 0x1b4`

5.6.1.132. `#define ALT_TAB 0x1e3`

5.6.1.133. `#define ALT_U 0x1b5`

5.6.1.134. `#define ALT_UP 0x1ea /* alt-up arrow */`

5.6.1.135. `#define ALT_V 0x1b6`

5.6.1.136. `#define ALT_W 0x1b7`

5.6.1.137. `#define ALT_X 0x1b8`

5.6.1.138. `#define ALT_Y 0x1b9`

5.6.1.139. `#define ALT_Z 0x1ba`

5.6.1.140. `#define ATR_MSK_ATTRIBUTES@ /* Obsolete */`

5.6.1.141. `#define ATR_NRM_NORMAL@ /* Obsolete */`

5.6.1.142. `#define BSDcurses 1 /* BSD Curses routines */`

5.6.1.143. `#define BUTTON1_CLICKED 0x00000004L`

5.6.1.144. `#define BUTTON1_DOUBLE_CLICKED 0x00000008L`

5.6.1.145. `#define BUTTON1_MOVED 0x00000010L /* PDCurses */`

5.6.1.146. `#define BUTTON1_PRESSED 0x00000002L`

5.6.1.147. `#define BUTTON1_RELEASED 0x00000001L`

5.6.1.148. `#define BUTTON1_TRIPLE_CLICKED 0x00000010L`

5.6.1.149. `#define BUTTON2_CLICKED 0x00000080L`

5.6.1.150. `#define BUTTON2_DOUBLE_CLICKED 0x00000100L`

```
5.6.1.151. #define BUTTON2_MOVED 0x00000200L /* PDCurses */
5.6.1.152. #define BUTTON2_PRESSED 0x00000040L
5.6.1.153. #define BUTTON2_RELEASED 0x00000020L
5.6.1.154. #define BUTTON2_TRIPLE_CLICKED 0x00000200L
5.6.1.155. #define BUTTON3_CLICKED 0x00001000L
5.6.1.156. #define BUTTON3_DOUBLE_CLICKED 0x00002000L
5.6.1.157. #define BUTTON3_MOVED 0x00004000L /* PDCurses */
5.6.1.158. #define BUTTON3_PRESSED 0x00000800L
5.6.1.159. #define BUTTON3_RELEASED 0x00000400L
5.6.1.160. #define BUTTON3_TRIPLE_CLICKED 0x00004000L
5.6.1.161. #define BUTTON4_CLICKED 0x00020000L
5.6.1.162. #define BUTTON4_DOUBLE_CLICKED 0x00040000L
5.6.1.163. #define BUTTON4_PRESSED 0x00010000L
5.6.1.164. #define BUTTON4_RELEASED 0x00008000L
5.6.1.165. #define BUTTON4_TRIPLE_CLICKED 0x00080000L
5.6.1.166. #define BUTTON5_CLICKED 0x00400000L
5.6.1.167. #define BUTTON5_DOUBLE_CLICKED 0x00800000L
5.6.1.168. #define BUTTON5_PRESSED 0x00200000L
5.6.1.169. #define BUTTON5_RELEASED 0x00100000L
5.6.1.170. #define BUTTON5_TRIPLE_CLICKED 0x01000000L
5.6.1.171. #define BUTTON_ACTION_MASK 0x0007 /* PDCurses */
5.6.1.172. #define BUTTON_ALTDC_BUTTON_ALT@
5.6.1.173. #define BUTTON_CHANGED( x ) (Mouse_status.changes & (1 << ((x) - 1)))
5.6.1.174. #define BUTTON_CLICKED 0x0002
5.6.1.175. #define BUTTON_CONTROLDC_BUTTON_CONTROL@
5.6.1.176. #define BUTTON_DOUBLE_CLICKED 0x0003
5.6.1.177. #define BUTTON_MODIFIER_ALT 0x10000000L /* PDCurses */
5.6.1.178. #define BUTTON_MODIFIER_CONTROL 0x08000000L /* PDCurses */
```

5.6.1.179. `#define BUTTON_MODIFIER_MASK 0x0038 /* PDCurses */`

5.6.1.180. `#define BUTTON_MODIFIER_SHIFT 0x04000000L /* PDCurses */`

5.6.1.181. `#define BUTTON_MOVED 0x0005 /* PDCurses */`

5.6.1.182. `#define BUTTON_PRESSED 0x0001`

5.6.1.183. `#define BUTTON_RELEASED 0x0000`

5.6.1.184. `#define BUTTON_SHIFTDC_BUTTON_SHIFT@`

5.6.1.185. `#define BUTTON_STATUS(x) (Mouse_status.button[(x) - 1])`

5.6.1.186. `#define BUTTON_TRIPLE_CLICKED 0x0004`

5.6.1.187. `#define CHR_MSK_CHARTEXT@ /* Obsolete */`

5.6.1.188. `#define CHTYPE_LONG 1 /* size of chtype; long */`

5.6.1.189. `#define COLOR_BLACK 0`

5.6.1.190. `#define COLOR_BLUE 1`

5.6.1.191. `#define COLOR_CYAN (COLOR_BLUE | COLOR_GREEN)`

5.6.1.192. `#define COLOR_GREEN 2`

5.6.1.193. `#define COLOR_MAGENTA (COLOR_RED | COLOR_BLUE)`

5.6.1.194. `#define COLOR_PAIR(n) (((chtype)(n) << PDC_COLOR_SHIFT) & A_COLOR)`

5.6.1.195. `#define COLOR_RED 4`

5.6.1.196. `#define COLOR_WHITE 7`

5.6.1.197. `#define COLOR_YELLOW (COLOR_RED | COLOR_GREEN)`

5.6.1.198. `#define CTL_BKSP 0x1f9 /* ctl-backspace */`

5.6.1.199. `#define CTL_DEL 0x20f /* clt-delete */`

5.6.1.200. `#define CTL_DOWN 0x1e1 /* ctl-down arrow */`

5.6.1.201. `#define CTL_END 0x1c0`

5.6.1.202. `#define CTL_ENTER 0x211 /* ctl-enter */`

5.6.1.203. `#define CTL_HOME 0x1bf`

5.6.1.204. `#define CTL_INS 0x1dd /* ctl-insert */`

5.6.1.205. `#define CTL_LEFT 0x1bb /* Control-Left-Arrow */`

5.6.1.206. `#define CTL_PAD0 0x1fb /* ctl-keypad 0 */`

```
5.6.1.207. #define CTL_PAD1 0x1fc
5.6.1.208. #define CTL_PAD2 0x1fd
5.6.1.209. #define CTL_PAD3 0x1fe
5.6.1.210. #define CTL_PAD4 0x1ff
5.6.1.211. #define CTL_PAD5 0x200
5.6.1.212. #define CTL_PAD6 0x201
5.6.1.213. #define CTL_PAD7 0x202
5.6.1.214. #define CTL_PAD8 0x203
5.6.1.215. #define CTL_PAD9 0x204
5.6.1.216. #define CTL_PADCENTER 0x1d3 /* ctl-enter on keypad */
5.6.1.217. #define CTL_PADENTER 0x1cc /* ctl-enter on keypad */
5.6.1.218. #define CTL_PADMINUS 0x1d5 /* ctl-minus on keypad */
5.6.1.219. #define CTL_PADPLUS 0x1d4 /* ctl-plus on keypad */
5.6.1.220. #define CTL_PADSLASH 0x1d6 /* ctl-slash on keypad */
5.6.1.221. #define CTL_PADSTAR 0x1d7 /* ctl-star on keypad */
5.6.1.222. #define CTL_PADSTOP 0x1d2 /* ctl-stop on keypad */
5.6.1.223. #define CTL_PGDN 0x1be
5.6.1.224. #define CTL_PGUP 0x1bd
5.6.1.225. #define CTL_RIGHT 0x1bc
5.6.1.226. #define CTL_TAB 0x1e2 /* ctl-tab */
5.6.1.227. #define CTL_UP 0x1e0 /* ctl-up arrow */
5.6.1.228. #define ERR (-1)
5.6.1.229. #define FALSE 0
5.6.1.230. #define getbegyx( w, y, x ) (y = getbegy(w), x = getbegx(w))
5.6.1.231. #define getch( ) wgetch(stdscr)
5.6.1.232. #define getmaxyx( w, y, x ) (y = getmaxy(w), x = getmaxx(w))
5.6.1.233. #define getparyx( w, y, x ) (y = getpary(w), x = getparx(w))
```

5.6.1.234. `#define getsyx(y, x)`

Valor:

```
{ if (curscr->_leaveit) (y)=(x)=-1; \
    else getyx(curscr, (y), (x)); }
```

5.6.1.235. `#define getyx(w, y, x) (y = getcury(w), x = getcurx(w))`

5.6.1.236. `#define KEY_A1 0x1c1 /* upper left on Virtual keypad */`

5.6.1.237. `#define KEY_A2 0x1c2 /* upper middle on Virt. keypad */`

5.6.1.238. `#define KEY_A3 0x1c3 /* upper right on Vir. keypad */`

5.6.1.239. `#define KEY_ABORT 0x15c /* abort/terminate key (any) */`

5.6.1.240. `#define KEY_ALT_L 0x220 /* Left-alt */`

5.6.1.241. `#define KEY_ALT_R 0x221 /* Right-alt */`

5.6.1.242. `#define KEY_B1 0x1c4 /* middle left on Virt. keypad */`

5.6.1.243. `#define KEY_B2 0x1c5 /* center on Virt. keypad */`

5.6.1.244. `#define KEY_B3 0x1c6 /* middle right on Vir. keypad */`

5.6.1.245. `#define KEY_BACKSPACE 0x107 /* not on pc */`

5.6.1.246. `#define KEY_BEG 0x160 /* beg(inning) key */`

5.6.1.247. `#define KEY_BREAK 0x101 /* Not on PC KBD */`

5.6.1.248. `#define KEY_BTAB 0x15f /* Back tab key */`

5.6.1.249. `#define KEY_C1 0x1c7 /* lower left on Virt. keypad */`

5.6.1.250. `#define KEY_C2 0x1c8 /* lower middle on Virt. keypad */`

5.6.1.251. `#define KEY_C3 0x1c9 /* lower right on Vir. keypad */`

5.6.1.252. `#define KEY_CANCEL 0x161 /* cancel key */`

5.6.1.253. `#define KEY_CATAB 0x156 /* clear all tabs */`

5.6.1.254. `#define KEY_CLEAR 0x14d /* clear screen */`

5.6.1.255. `#define KEY_CLOSE 0x162 /* close key */`

5.6.1.256. `#define KEY_CODE_YES 0x100 /* If get_wch() gives a key code */`

5.6.1.257. `#define KEY_COMMAND 0x163 /* cmd (command) key */`

5.6.1.258. `#define KEY_CONTROL_L 0x21e /* Left-control */`

5.6.1.259. `#define KEY_CONTROL_R 0x21f /* Right-control */`

5.6.1.260. `#define KEY_COPY 0x164 /* copy key */`

5.6.1.261. `#define KEY_CREATE 0x165 /* create key */`

5.6.1.262. `#define KEY_CTAB 0x155 /* clear tab */`

5.6.1.263. `#define KEY_DC 0x14a /* delete character */`

5.6.1.264. `#define KEY_DL 0x148 /* delete line */`

5.6.1.265. `#define KEY_DOWN 0x102 /* Down arrow key */`

5.6.1.266. `#define KEY_EIC 0x14c /* exit insert char mode */`

5.6.1.267. `#define KEY_END 0x166 /* end key */`

5.6.1.268. `#define KEY_ENTER 0x157 /* enter or send (unreliable) */`

5.6.1.269. `#define KEY_EOL 0x14f /* clear to end of line */`

5.6.1.270. `#define KEY_EOS 0x14e /* clear to end of screen */`

5.6.1.271. `#define KEY_EXIT 0x167 /* exit key */`

5.6.1.272. `#define KEY_F(n)(KEY_F0 + (n))`

5.6.1.273. `#define KEY_F0 0x108 /* function keys; 64 reserved */`

5.6.1.274. `#define KEY_FIND 0x168 /* find key */`

5.6.1.275. `#define KEY_HELP 0x169 /* help key */`

5.6.1.276. `#define KEY_HOME 0x106 /* home key */`

5.6.1.277. `#define KEY_IC 0x14b /* insert char or enter ins mode */`

5.6.1.278. `#define KEY_IL 0x149 /* insert line */`

5.6.1.279. `#define KEY_LEFT 0x104 /* Left arrow key */`

5.6.1.280. `#define KEY_LHELP 0x15e /* long help */`

5.6.1.281. `#define KEY_LL 0x15b /* home down/bottom (lower left) */`

5.6.1.282. `#define KEY_MARK 0x16a /* mark key */`

5.6.1.283. `#define KEY_MAXKEY_SDOWN@ /* Maximum curses key */`

5.6.1.284. `#define KEY_MESSAGE 0x16b /* message key */`

5.6.1.285. `#define KEY_MINEY_BREAK@ /* Minimum curses key value */`

5.6.1.286. `#define KEY_MOUSE 0x21b /* "mouse" key */`

5.6.1.287. `#define KEY_MOVE 0x16c /* move key */`

5.6.1.288. `#define KEY_NEXT 0x16d /* next object key */`

5.6.1.289. `#define KEY_NPAGE 0x152 /* next page */`

5.6.1.290. `#define KEY_OPEN 0x16e /* open key */`

5.6.1.291. `#define KEY_OPTIONS 0x16f /* options key */`

5.6.1.292. `#define KEY_PPAGE 0x153 /* previous page */`

5.6.1.293. `#define KEY_PREVIOUS 0x170 /* previous object key */`

5.6.1.294. `#define KEY_PRINT 0x15a /* print/copy */`

5.6.1.295. `#define KEY_REDO 0x171 /* redo key */`

5.6.1.296. `#define KEY_REFERENCE 0x172 /* ref(ERENCE) key */`

5.6.1.297. `#define KEY_REFRESH 0x173 /* refresh key */`

5.6.1.298. `#define KEY_REPLACE 0x174 /* replace key */`

5.6.1.299. `#define KEY_RESET 0x159 /* reset/hard reset (unreliable) */`

5.6.1.300. `#define KEY_RESIZE 0x222 /* Window resize */`

5.6.1.301. `#define KEY_RESTART 0x175 /* restart key */`

5.6.1.302. `#define KEY_RESUME 0x176 /* resume key */`

5.6.1.303. `#define KEY_RIGHT 0x105 /* Right arrow key */`

5.6.1.304. `#define KEY_SAVE 0x177 /* save key */`

5.6.1.305. `#define KEY_SBEG 0x178 /* shifted beginning key */`

5.6.1.306. `#define KEY_SCANCEL 0x179 /* shifted cancel key */`

5.6.1.307. `#define KEY_SCOMMAND 0x17a /* shifted command key */`

5.6.1.308. `#define KEY_SCOPY 0x17b /* shifted copy key */`

5.6.1.309. `#define KEY_SCREATE 0x17c /* shifted create key */`

5.6.1.310. `#define KEY_SDC 0x17d /* shifted delete char key */`

5.6.1.311. `#define KEY_SDL 0x17e /* shifted delete line key */`

5.6.1.312. `#define KEY_SDOWN 0x224 /* Shifted down arrow */`

5.6.1.313. `#define KEY_SELECT 0x17f /* select key */`

5.6.1.314. `#define KEY_SEND 0x180 /* shifted end key */`

5.6.1.315. `#define KEY_SEOL 0x181 /* shifted clear line key */`

5.6.1.316. `#define KEY_SEXIT 0x182 /* shifted exit key */`

5.6.1.317. `#define KEY_SF 0x150 /* scroll 1 line forward */`

5.6.1.318. `#define KEY_SFIND 0x183 /* shifted find key */`

5.6.1.319. `#define KEY_SHELP 0x15d /* short help */`

5.6.1.320. `#define KEY_SHIFT_L 0x21c /* Left-shift */`

5.6.1.321. `#define KEY_SHIFT_R 0x21d /* Right-shift */`

5.6.1.322. `#define KEY_SHOME 0x184 /* shifted home key */`

5.6.1.323. `#define KEY_SIC 0x185 /* shifted input key */`

5.6.1.324. `#define KEY_SLEFT 0x187 /* shifted left arrow key */`

5.6.1.325. `#define KEY_SMESSAGE 0x188 /* shifted message key */`

5.6.1.326. `#define KEY_SMOVE 0x189 /* shifted move key */`

5.6.1.327. `#define KEY_SNEXT 0x18a /* shifted next key */`

5.6.1.328. `#define KEY_SOPTIONS 0x18b /* shifted options key */`

5.6.1.329. `#define KEY_SPREVIOUS 0x18c /* shifted prev key */`

5.6.1.330. `#define KEY_SPRINT 0x18d /* shifted print key */`

5.6.1.331. `#define KEY_SR 0x151 /* scroll 1 line back (reverse) */`

5.6.1.332. `#define KEY_SREDO 0x18e /* shifted redo key */`

5.6.1.333. `#define KEY_SREPLACE 0x18f /* shifted replace key */`

5.6.1.334. `#define KEY_SRESET 0x158 /* soft/reset (partial/unreliable) */`

5.6.1.335. `#define KEY_SRIGHT 0x190 /* shifted right arrow */`

5.6.1.336. `#define KEY_SRSUME 0x191 /* shifted resume key */`

5.6.1.337. `#define KEY_SSAVE 0x192 /* shifted save key */`

5.6.1.338. `#define KEY_SSUSPEND 0x193 /* shifted suspend key */`

5.6.1.339. `#define KEY_STAB 0x154 /* set tab */`

5.6.1.340. `#define KEY_SUNDO 0x194 /* shifted undo key */`

5.6.1.341. `#define KEY_SUP 0x223 /* Shifted up arrow */`

5.6.1.342. `#define KEY_SUSPEND 0x195 /* suspend key */`

5.6.1.343. `#define KEY_UNDO 0x196 /* undo key */`

5.6.1.344. `#define KEY_UP 0x103 /* Up arrow key */`

5.6.1.345. `#define MOUSE_MOVED (Mouse_status.changes & PDC_MOUSE_MOVED)`

5.6.1.346. `#define MOUSE_POS_REPORT (Mouse_status.changes & PDC_MOUSE_POSITION)`

5.6.1.347. `#define MOUSE_WHEEL_DOWN (Mouse_status.changes & PDC_MOUSE_WHEEL_DOWN)`

5.6.1.348. `#define MOUSE_WHEEL_SCROLL 0x02000000L /* PDCurses */`

5.6.1.349. `#define MOUSE_WHEEL_UP (Mouse_status.changes & PDC_MOUSE_WHEEL_UP)`

5.6.1.350. `#define MOUSE_X_POS (Mouse_status.x)`

5.6.1.351. `#define MOUSE_Y_POS (Mouse_status.y)`

5.6.1.352. `#define NULL (void *)0`

5.6.1.353. `#define OK 0`

5.6.1.354. `#define PAD0 0x1fa /* keypad 0 */`

5.6.1.355. `#define PADENTER 0x1cb /* enter on keypad */`

5.6.1.356. `#define PADMINUS 0x1d0 /* minus on keypad */`

5.6.1.357. `#define PADPLUS 0x1d1 /* plus on keypad */`

5.6.1.358. `#define PADSLASH 0x1ca /* slash on keypad */`

5.6.1.359. `#define PADSTAR 0x1cf /* star on keypad */`

5.6.1.360. `#define PADSTOP 0x1ce /* stop on keypad */`

5.6.1.361. `#define PAIR_NUMBER(n) (((n) & A_COLOR) >> PDC_COLOR_SHIFT)`

5.6.1.362. `#define PDC_ATTR_SHIFT 19`

5.6.1.363. `#define PDC_BUILD 3401`

5.6.1.364. `#define PDC_BUTTON_ALT 0x0020 /* PDCurses */`

5.6.1.365. `#define PDC_BUTTON_CONTROL 0x0010 /* PDCurses */`

5.6.1.366. `#define PDC_BUTTON_SHIFT 0x0008 /* PDCurses */`

5.6.1.367. `#define PDC_CLIP_ACCESS_ERROR 1`

5.6.1.368. `#define PDC_CLIP_EMPTY 2`

5.6.1.369. `#define PDC_CLIP_MEMORY_ERROR 3`

5.6.1.370. `#define PDC_CLIP_SUCCESS 0`

5.6.1.371. `#define PDC_COLOR_SHIFT 24`

5.6.1.372. `#define PDC_KEY_MODIFIER_ALT 4`

5.6.1.373. `#define PDC_KEY_MODIFIER_CONTROL 2`

5.6.1.374. `#define PDC_KEY_MODIFIER_NUMLOCK 8`

5.6.1.375. `#define PDC_KEY_MODIFIER_SHIFT 1`

5.6.1.376. `#define PDC_MOUSE_MOVED 0x0008`

5.6.1.377. `#define PDC_MOUSE_POSITION 0x0010`

5.6.1.378. `#define PDC_MOUSE_WHEEL_DOWN 0x0040`

5.6.1.379. `#define PDC_MOUSE_WHEEL_UP 0x0020`

5.6.1.380. `#define PDCEX extern`

5.6.1.381. `#define PDCURSES 1 /* PDCurses-only routines */`

5.6.1.382. `#define REPORT_MOUSE_POSITION 0x20000000L`

5.6.1.383. `#define SHF_DC 0x21a /* shift-delete on keypad */`

5.6.1.384. `#define SHF_DOWN 0x218 /* shift-down on keypad */`

5.6.1.385. `#define SHF_IC 0x219 /* shift-insert on keypad */`

5.6.1.386. `#define SHF_PADENTER 0x212 /* shift-enter on keypad */`

5.6.1.387. `#define SHF_PADMINUS 0x216 /* shift-minus on keypad */`

5.6.1.388. `#define SHF_PADPLUS 0x215 /* shift-plus on keypad */`

5.6.1.389. `#define SHF_PADSLASH 0x213 /* shift-slash on keypad */`

5.6.1.390. `#define SHF_PADSTAR 0x214 /* shift-star on keypad */`

5.6.1.391. `#define SHF_UP 0x217 /* shift-up on keypad */`

5.6.1.392. `#define SYSVcurses 1 /* System V Curses routines */`

5.6.1.393. `#define TRUE 1`

5.6.1.394. `#define ungetch(ch)DC_ungetch@(ch)`

5.6.1.395. `#define WA_ALTCHARSET_ALTCHARSET@`

5.6.1.396. `#define WA_BLINK_BLINK@`

5.6.1.397. `#define WA_BOLD_BOLD@`

5.6.1.398. `#define WA_DIM_DIM@`

5.6.1.399. `#define WA_HORIZONTAL_NORMAL@`

5.6.1.400. `#define WA_INVIS_INVIS@`

5.6.1.401. `#define WA_LEFT_LEFTLINE@`

5.6.1.402. `#define WA_LOW_NORMAL@`

5.6.1.403. `#define WA_PROTECT_PROTECT@`

5.6.1.404. `#define WA_REVERSE_REVERSE@`

5.6.1.405. `#define WA_RIGHT_RIGHTLINE@`

5.6.1.406. `#define WA_STANDOUT_STANDOUT@`

5.6.1.407. `#define WA_TOP_NORMAL@`

5.6.1.408. `#define WA_UNDERLINE_UNDERLINE@`

5.6.1.409. `#define WA_VERTICAL_NORMAL@`

5.6.1.410. `#define WHEEL_SCROLLED 0x0006 /* PDCurses */`

5.6.1.411. `#define XOPEN 1 /* X/Open Curses routines */`

5.6.2. Documentación de los 'typedefs'

5.6.2.1. `typedef chtype attr_t`

5.6.2.2. `typedef unsigned char bool`

5.6.2.3. `typedef unsigned long chtype`

5.6.2.4. `typedef unsigned long mmask_t`

5.6.2.5. `typedef struct _win WINDOW`

5.6.3. Documentación de las funciones

5.6.3.1. `int addch (const chtype)`

5.6.3.2. `int addchnstr (const chtype *, int)`

5.6.3.3. `int addchstr (const chtype *)`

5.6.3.4. `int addnstr (const char *, int)`

5.6.3.5. `int addrawch (chtype)`

5.6.3.6. `int addstr (const char *)`

5.6.3.7. `int assume_default_colors (int , int)`

5.6.3.8. `int attr_get (attr_t *, short *, void *)`

- 5.6.3.9. `int attr_off (attr_t , void *)`
- 5.6.3.10. `int attr_on (attr_t , void *)`
- 5.6.3.11. `int attr_set (attr_t , short , void *)`
- 5.6.3.12. `int attroff (chtype)`
- 5.6.3.13. `int attron (chtype)`
- 5.6.3.14. `int attrset (chtype)`
- 5.6.3.15. `int baudrate (void)`
- 5.6.3.16. `int beep (void)`
- 5.6.3.17. `int bkgd (chtype)`
- 5.6.3.18. `void bkgdset (chtype)`
- 5.6.3.19. `int border (chtype , chtype , chtype , chtype , chtype , chtype , chtype , chtype)`
- 5.6.3.20. `int box (WINDOW * , chtype , chtype)`
- 5.6.3.21. `bool can_change_color (void)`
- 5.6.3.22. `int cbreak (void)`
- 5.6.3.23. `int chgat (int , attr_t , short , const void *)`
- 5.6.3.24. `int clear (void)`
- 5.6.3.25. `int clearok (WINDOW * , bool)`
- 5.6.3.26. `int clrtoebot (void)`
- 5.6.3.27. `int clrtoeol (void)`
- 5.6.3.28. `int color_content (short , short * , short * , short *)`
- 5.6.3.29. `int color_set (short , void *)`
- 5.6.3.30. `int copywin (const WINDOW * , WINDOW * , int , int , int , int , int , int , int)`
- 5.6.3.31. `int crmode (void)`
- 5.6.3.32. `int curs_set (int)`
- 5.6.3.33. `const char* curses_version (void)`
- 5.6.3.34. `int def_prog_mode (void)`
- 5.6.3.35. `int def_shell_mode (void)`
- 5.6.3.36. `int delay_output (int)`

- 5.6.3.37. int delch (void)
- 5.6.3.38. int deleteln (void)
- 5.6.3.39. void delscreen (SCREEN *)
- 5.6.3.40. int delwin (WINDOW *)
- 5.6.3.41. WINDOW* derwin (WINDOW * , int , int , int , int)
- 5.6.3.42. int doupdate (void)
- 5.6.3.43. int draino (int)
- 5.6.3.44. WINDOW* dupwin (WINDOW *)
- 5.6.3.45. int echo (void)
- 5.6.3.46. int echochar (const chtype)
- 5.6.3.47. int endwin (void)
- 5.6.3.48. int erase (void)
- 5.6.3.49. char erasechar (void)
- 5.6.3.50. void filter (void)
- 5.6.3.51. int fixterm (void)
- 5.6.3.52. int flash (void)
- 5.6.3.53. int flushinp (void)
- 5.6.3.54. chtype getattrs (WINDOW *)
- 5.6.3.55. int getbegx (WINDOW *)
- 5.6.3.56. int getbegy (WINDOW *)
- 5.6.3.57. chtype getbkgd (WINDOW *)
- 5.6.3.58. unsigned long getbmap (void)
- 5.6.3.59. int getcurx (WINDOW *)
- 5.6.3.60. int getcury (WINDOW *)
- 5.6.3.61. int getmaxx (WINDOW *)
- 5.6.3.62. int getmaxy (WINDOW *)
- 5.6.3.63. unsigned long getmouse (void)
- 5.6.3.64. int getnstr (char * , int)

5.6.3.65. `int getparx (WINDOW *)`

5.6.3.66. `int getpary (WINDOW *)`

5.6.3.67. `int getstr (char *)`

5.6.3.68. `WINDOW* getwin (FILE *)`

5.6.3.69. `int halfdelay (int)`

5.6.3.70. `bool has_colors (void)`

5.6.3.71. `bool has_ic (void)`

5.6.3.72. `bool has_il (void)`

5.6.3.73. `bool has_key (int)`

5.6.3.74. `int hline (chtype , int)`

5.6.3.75. `void idcok (WINDOW * , bool)`

5.6.3.76. `int idlok (WINDOW * , bool)`

5.6.3.77. `void immedok (WINDOW * , bool)`

5.6.3.78. `chtype inch (void)`

5.6.3.79. `int inchnstr (chtype * , int)`

5.6.3.80. `int inchstr (chtype *)`

5.6.3.81. `int init_color (short , short , short , short)`

5.6.3.82. `int init_pair (short , short , short)`

5.6.3.83. `WINDOW* initscr (void)`

5.6.3.84. `int innstr (char * , int)`

5.6.3.85. `int insch (chtype)`

5.6.3.86. `int insdelln (int)`

5.6.3.87. `int insertln (void)`

5.6.3.88. `int insnstr (const char * , int)`

5.6.3.89. `int insrawch (chtype)`

5.6.3.90. `int insstr (const char *)`

5.6.3.91. `int instr (char *)`

5.6.3.92. `int intrflush (WINDOW * , bool)`

- 5.6.3.93. `bool is_linetouched (WINDOW *, int)`
- 5.6.3.94. `bool is_termresized (void)`
- 5.6.3.95. `bool is_wintouched (WINDOW *)`
- 5.6.3.96. `bool isendwin (void)`
- 5.6.3.97. `char* keyname (int)`
- 5.6.3.98. `int keypad (WINDOW *, bool)`
- 5.6.3.99. `char killchar (void)`
- 5.6.3.100. `int leaveok (WINDOW *, bool)`
- 5.6.3.101. `char* longname (void)`
- 5.6.3.102. `int map_button (unsigned long)`
- 5.6.3.103. `int meta (WINDOW *, bool)`
- 5.6.3.104. `int mouse_off (unsigned long)`
- 5.6.3.105. `int mouse_on (unsigned long)`
- 5.6.3.106. `int mouse_set (unsigned long)`
- 5.6.3.107. `bool mouse_trafo (int *, int *, bool)`
- 5.6.3.108. `int mouseinterval (int)`
- 5.6.3.109. `mmask_t mousemask (mmask_t, mmask_t *)`
- 5.6.3.110. `int move (int, int)`
- 5.6.3.111. `int mvaddch (int, int, const chtype)`
- 5.6.3.112. `int mvaddchnstr (int, int, const chtype *, int)`
- 5.6.3.113. `int mvaddchstr (int, int, const chtype *)`
- 5.6.3.114. `int mvaddnstr (int, int, const char *, int)`
- 5.6.3.115. `int mvaddrawch (int, int, chtype)`
- 5.6.3.116. `int mvaddstr (int, int, const char *)`
- 5.6.3.117. `int mvchgat (int, int, int, attr_t, short, const void *)`
- 5.6.3.118. `int mvcur (int, int, int, int)`
- 5.6.3.119. `int mvdelch (int, int)`
- 5.6.3.120. `int mvdeleteln (int, int)`

- 5.6.3.121. `int mvderwin (WINDOW *, int, int)`
- 5.6.3.122. `int mvgetch (int, int)`
- 5.6.3.123. `int mvgetnstr (int, int, char *, int)`
- 5.6.3.124. `int mvgetstr (int, int, char *)`
- 5.6.3.125. `int mvhline (int, int, chtype, int)`
- 5.6.3.126. `chtype mvinch (int, int)`
- 5.6.3.127. `int mvinchnstr (int, int, chtype *, int)`
- 5.6.3.128. `int mvinchstr (int, int, chtype *)`
- 5.6.3.129. `int mvinnstr (int, int, char *, int)`
- 5.6.3.130. `int mvinsch (int, int, chtype)`
- 5.6.3.131. `int mvinsertln (int, int)`
- 5.6.3.132. `int mvinsnstr (int, int, const char *, int)`
- 5.6.3.133. `int mvinsrawch (int, int, chtype)`
- 5.6.3.134. `int mvinsstr (int, int, const char *)`
- 5.6.3.135. `int mvinstr (int, int, char *)`
- 5.6.3.136. `int mvprintw (int, int, const char *, ...)`
- 5.6.3.137. `int mvscanw (int, int, const char *, ...)`
- 5.6.3.138. `int mvvline (int, int, chtype, int)`
- 5.6.3.139. `int mvwaddch (WINDOW *, int, int, const chtype)`
- 5.6.3.140. `int mvwaddchnstr (WINDOW *, int, int, const chtype *, int)`
- 5.6.3.141. `int mvwaddchstr (WINDOW *, int, int, const chtype *)`
- 5.6.3.142. `int mvwaddnstr (WINDOW *, int, int, const char *, int)`
- 5.6.3.143. `int mvwaddrawch (WINDOW *, int, int, chtype)`
- 5.6.3.144. `int mvwaddstr (WINDOW *, int, int, const char *)`
- 5.6.3.145. `int mvwchgat (WINDOW *, int, int, int, attr_t, short, const void *)`
- 5.6.3.146. `int mvwdelch (WINDOW *, int, int)`
- 5.6.3.147. `int mvwdeleteln (WINDOW *, int, int)`
- 5.6.3.148. `int mvwgetch (WINDOW *, int, int)`

- 5.6.3.149. `int mvwgetnstr (WINDOW *, int, int, char *, int)`
- 5.6.3.150. `int mvwgetstr (WINDOW *, int, int, char *)`
- 5.6.3.151. `int mvwhtline (WINDOW *, int, int, chtype, int)`
- 5.6.3.152. `int mvwin (WINDOW *, int, int)`
- 5.6.3.153. `chtype mvwinch (WINDOW *, int, int)`
- 5.6.3.154. `int mvwinchnstr (WINDOW *, int, int, chtype *, int)`
- 5.6.3.155. `int mvwinchstr (WINDOW *, int, int, chtype *)`
- 5.6.3.156. `int mvwinstr (WINDOW *, int, int, char *, int)`
- 5.6.3.157. `int mvwinsch (WINDOW *, int, int, chtype)`
- 5.6.3.158. `int mvwininsertln (WINDOW *, int, int)`
- 5.6.3.159. `int mvwinsnstr (WINDOW *, int, int, const char *, int)`
- 5.6.3.160. `int mvwinsrawch (WINDOW *, int, int, chtype)`
- 5.6.3.161. `int mvwinsstr (WINDOW *, int, int, const char *)`
- 5.6.3.162. `int mvwinstr (WINDOW *, int, int, char *)`
- 5.6.3.163. `int mvwprintw (WINDOW *, int, int, const char *, ...)`
- 5.6.3.164. `int mvwscanw (WINDOW *, int, int, const char *, ...)`
- 5.6.3.165. `int mvwvline (WINDOW *, int, int, chtype, int)`
- 5.6.3.166. `int napms (int)`
- 5.6.3.167. `int nc_getmouse (MEVENT *)`
- 5.6.3.168. `WINDOW* newpad (int, int)`
- 5.6.3.169. `SCREEN* newterm (const char *, FILE *, FILE *)`
- 5.6.3.170. `WINDOW* newwin (int, int, int, int)`
- 5.6.3.171. `int nl (void)`
- 5.6.3.172. `int nocbreak (void)`
- 5.6.3.173. `int nocrmode (void)`
- 5.6.3.174. `int nodelay (WINDOW *, bool)`
- 5.6.3.175. `int noecho (void)`
- 5.6.3.176. `int nonl (void)`

- 5.6.3.177. void noqiflush (void)
- 5.6.3.178. int noraw (void)
- 5.6.3.179. int notimeout (WINDOW *, bool)
- 5.6.3.180. int overlay (const WINDOW *, WINDOW *)
- 5.6.3.181. int overwrite (const WINDOW *, WINDOW *)
- 5.6.3.182. int pair_content (short , short * , short *)
- 5.6.3.183. int PDC_clearclipboard (void)
- 5.6.3.184. void PDC_debug (const char * , ...)
- 5.6.3.185. int PDC_freeclipboard (char *)
- 5.6.3.186. unsigned long PDC_get_input_fd (void)
- 5.6.3.187. unsigned long PDC_get_key_modifiers (void)
- 5.6.3.188. int PDC_getclipboard (char **, long *)
- 5.6.3.189. int PDC_return_key_modifiers (bool)
- 5.6.3.190. int PDC_save_key_modifiers (bool)
- 5.6.3.191. int PDC_set_blink (bool)
- 5.6.3.192. int PDC_set_line_color (short)
- 5.6.3.193. void PDC_set_title (const char *)
- 5.6.3.194. int PDC_setclipboard (const char * , long)
- 5.6.3.195. int PDC_ungetch (int)
- 5.6.3.196. int pechochar (WINDOW * , chtype)
- 5.6.3.197. int pnoutrefresh (WINDOW * , int , int , int , int , int , int)
- 5.6.3.198. int prefresh (WINDOW * , int , int , int , int , int , int)
- 5.6.3.199. int printw (const char * , ...)
- 5.6.3.200. int putwin (WINDOW * , FILE *)
- 5.6.3.201. void qiflush (void)
- 5.6.3.202. int raw (void)
- 5.6.3.203. int raw_output (bool)
- 5.6.3.204. int redrawwin (WINDOW *)

- 5.6.3.205. int refresh (void)
- 5.6.3.206. int request_mouse_pos (void)
- 5.6.3.207. int reset_prog_mode (void)
- 5.6.3.208. int reset_shell_mode (void)
- 5.6.3.209. int resetterm (void)
- 5.6.3.210. int resetty (void)
- 5.6.3.211. int resize_term (int , int)
- 5.6.3.212. WINDOW* resize_window (WINDOW *, int , int)
- 5.6.3.213. int ripoffline (int , int(*) (WINDOW *, int))
- 5.6.3.214. int saveterm (void)
- 5.6.3.215. int savetty (void)
- 5.6.3.216. int scanw (const char *, ...)
- 5.6.3.217. int scr_dump (const char *)
- 5.6.3.218. int scr_init (const char *)
- 5.6.3.219. int scr_restore (const char *)
- 5.6.3.220. int scr_set (const char *)
- 5.6.3.221. int sclr (int)
- 5.6.3.222. int scroll (WINDOW *)
- 5.6.3.223. int scrollok (WINDOW *, bool)
- 5.6.3.224. SCREEN* set_term (SCREEN *)
- 5.6.3.225. int setscrreg (int , int)
- 5.6.3.226. int setsyx (int , int)
- 5.6.3.227. int slk_attr_off (const attr_t , void *)
- 5.6.3.228. int slk_attr_on (const attr_t , void *)
- 5.6.3.229. int slk_attr_set (const attr_t , short , void *)
- 5.6.3.230. int slk_attroff (const chtype)
- 5.6.3.231. int slk_attron (const chtype)
- 5.6.3.232. int slk_attrset (const chtype)

5.6.3.233. int slk_clear (void)

5.6.3.234. int slk_color (short)

5.6.3.235. int slk_init (int)

5.6.3.236. char* slk_label (int)

5.6.3.237. int slk_noutrefresh (void)

5.6.3.238. int slk_refresh (void)

5.6.3.239. int slk_restore (void)

5.6.3.240. int slk_set (int , const char * , int)

5.6.3.241. int slk_touch (void)

5.6.3.242. int standend (void)

5.6.3.243. int standout (void)

5.6.3.244. int start_color (void)

5.6.3.245. WINDOW* subpad (WINDOW * , int , int , int , int)

5.6.3.246. WINDOW* subwin (WINDOW * , int , int , int , int)

5.6.3.247. int syncok (WINDOW * , bool)

5.6.3.248. attr_t term_attrs (void)

5.6.3.249. chtype termattrs (void)

5.6.3.250. char* termname (void)

5.6.3.251. void timeout (int)

5.6.3.252. int touchline (WINDOW * , int , int)

5.6.3.253. int touchwin (WINDOW *)

5.6.3.254. void traceoff (void)

5.6.3.255. void traceon (void)

5.6.3.256. int typeahead (int)

5.6.3.257. char* unctrl (chtype)

5.6.3.258. int ungetmouse (MEVENT *)

5.6.3.259. int untouchwin (WINDOW *)

5.6.3.260. int use_default_colors (void)

- 5.6.3.261. void use_env (bool)
- 5.6.3.262. int vid_attr (attr_t, short, void *)
- 5.6.3.263. int vid_puts (attr_t, short, void *, int (*)(int))
- 5.6.3.264. int vidattr (chtype)
- 5.6.3.265. int vidputs (chtype, int (*)(int))
- 5.6.3.266. int vline (chtype, int)
- 5.6.3.267. int vw_printw (WINDOW *, const char *, va_list)
- 5.6.3.268. int vw_scanw (WINDOW *, const char *, va_list)
- 5.6.3.269. int vwprintw (WINDOW *, const char *, va_list)
- 5.6.3.270. int vwscanw (WINDOW *, const char *, va_list)
- 5.6.3.271. int waddch (WINDOW *, const chtype)
- 5.6.3.272. int waddchnstr (WINDOW *, const chtype *, int)
- 5.6.3.273. int waddchstr (WINDOW *, const chtype *)
- 5.6.3.274. int waddnstr (WINDOW *, const char *, int)
- 5.6.3.275. int waddrawch (WINDOW *, chtype)
- 5.6.3.276. int waddstr (WINDOW *, const char *)
- 5.6.3.277. int wattr_get (WINDOW *, attr_t *, short *, void *)
- 5.6.3.278. int wattr_off (WINDOW *, attr_t, void *)
- 5.6.3.279. int wattr_on (WINDOW *, attr_t, void *)
- 5.6.3.280. int wattr_set (WINDOW *, attr_t, short, void *)
- 5.6.3.281. int wattroff (WINDOW *, chtype)
- 5.6.3.282. int wattron (WINDOW *, chtype)
- 5.6.3.283. int wattrset (WINDOW *, chtype)
- 5.6.3.284. int wbkgd (WINDOW *, chtype)
- 5.6.3.285. void wbkgdset (WINDOW *, chtype)
- 5.6.3.286. int wborder (WINDOW *, chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype)
- 5.6.3.287. int wchgat (WINDOW *, int, attr_t, short, const void *)
- 5.6.3.288. int wclear (WINDOW *)

- 5.6.3.289. `int wclrtobot (WINDOW *)`
- 5.6.3.290. `int wclrtoeol (WINDOW *)`
- 5.6.3.291. `int wcolor_set (WINDOW *, short , void *)`
- 5.6.3.292. `void wcursyncup (WINDOW *)`
- 5.6.3.293. `int wdelch (WINDOW *)`
- 5.6.3.294. `int wdeleteln (WINDOW *)`
- 5.6.3.295. `int wechochar (WINDOW *, const chtype)`
- 5.6.3.296. `bool wenclose (const WINDOW *, int , int)`
- 5.6.3.297. `int werase (WINDOW *)`
- 5.6.3.298. `int wgetch (WINDOW *)`
- 5.6.3.299. `int wgetnstr (WINDOW *, char *, int)`
- 5.6.3.300. `int wgetstr (WINDOW *, char *)`
- 5.6.3.301. `int whline (WINDOW *, chtype , int)`
- 5.6.3.302. `chtype winch (WINDOW *)`
- 5.6.3.303. `int winchnstr (WINDOW *, chtype *, int)`
- 5.6.3.304. `int winchstr (WINDOW *, chtype *)`
- 5.6.3.305. `int winnstr (WINDOW *, char *, int)`
- 5.6.3.306. `int winsch (WINDOW *, chtype)`
- 5.6.3.307. `int winsdelln (WINDOW *, int)`
- 5.6.3.308. `int winsertln (WINDOW *)`
- 5.6.3.309. `int winsnstr (WINDOW *, const char *, int)`
- 5.6.3.310. `int winsrawch (WINDOW *, chtype)`
- 5.6.3.311. `int winsstr (WINDOW *, const char *)`
- 5.6.3.312. `int winstr (WINDOW *, char *)`
- 5.6.3.313. `void wmouse_position (WINDOW *, int *, int *)`
- 5.6.3.314. `bool wmouse_trafo (const WINDOW *, int *, int *, bool)`
- 5.6.3.315. `int wmove (WINDOW *, int , int)`
- 5.6.3.316. `int wnoutrefresh (WINDOW *)`

- 5.6.3.317. char wordchar (void)
- 5.6.3.318. int wprintw (WINDOW *, const char *, ...)
- 5.6.3.319. int wredrawln (WINDOW *, int, int)
- 5.6.3.320. int wrefresh (WINDOW *)
- 5.6.3.321. int wresize (WINDOW *, int, int)
- 5.6.3.322. int wscanw (WINDOW *, const char *, ...)
- 5.6.3.323. int wscrln (WINDOW *, int)
- 5.6.3.324. int wsetscrreg (WINDOW *, int, int)
- 5.6.3.325. int wstandend (WINDOW *)
- 5.6.3.326. int wstandout (WINDOW *)
- 5.6.3.327. void wsyncdown (WINDOW *)
- 5.6.3.328. void wsyncup (WINDOW *)
- 5.6.3.329. void wtimeout (WINDOW *, int)
- 5.6.3.330. int wtouchln (WINDOW *, int, int, int)
- 5.6.3.331. int wvline (WINDOW *, chtype, int)

5.6.4. Documentación de las variables

- 5.6.4.1. PDCEX chtype acs_map[]
- 5.6.4.2. PDCEX int COLOR_PAIRS
- 5.6.4.3. PDCEX int COLORS
- 5.6.4.4. PDCEX int COLS
- 5.6.4.5. PDCEX WINDOW* curscr
- 5.6.4.6. PDCEX int LINES
- 5.6.4.7. PDCEX MOUSE_STATUS Mouse_status
- 5.6.4.8. PDCEX SCREEN* SP
- 5.6.4.9. PDCEX WINDOW* stdscr
- 5.6.4.10. PDCEX int TABSIZE
- 5.6.4.11. PDCEX char ttytype[]

5.7. Referencia del Archivo decoder.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "mostrar.h"
#include "ALU.h"
#include "curses.h"
#include "desplazamiento.h"
#include "branch.h"
#include "decoder.h"
```

Funciones

- void [decodeInstruction](#) ([instruction_t](#) instruction, uint32_t *Registro, char *R_Banderas)
Decodifica la instrucción y la ejecuta.
- [instruction_t getInstruction](#) (char *instStr)
Obtiene la instrucción separada por partes.
- int [readFile](#) (char *filename, [ins_t](#) *instructions)
- int [countLines](#) (FILE *fp)

5.7.1. Documentación de las funciones

5.7.1.1. int countLines (FILE * fp)

5.7.1.2. void decodeInstruction (instruction_t instruction, uint32_t * Registro, char * R_Banderas)

Decodifica la instrucción y la ejecuta.

Parámetros

<i>instruction</i>	instrucción a decodificar y ejecutar
<i>*Registro</i>	Puntero al registro
<i>*R_Banderas</i>	Puntero a las banderas

5.7.1.3. instruction_t getInstruction (char * instStr)

Obtiene la instrucción separada por partes.

Parámetros

<i>instStr</i>	cadena que contiene la instrucción
----------------	------------------------------------

Devuelve

[instruction_t](#) la instrucción separada por partes

5.7.1.4. int readFile (char * filename, ins_t * instructions)

5.8. Referencia del Archivo decoder.h

```
#include <stdlib.h>
```

```
#include <stdio.h>
#include <string.h>
#include <stdint.h>
```

Estructuras de datos

- struct [ins_t](#)
- struct [instruction_t](#)

Funciones

- void [decodeInstruction](#) ([instruction_t](#) instruction, uint32_t *Registro, char *R_Banderas)
Decodifica la instrucción y la ejecuta.
- [instruction_t](#) [getInstruction](#) (char *instStr)
Obtiene la instrucción separada por partes.
- int [readFile](#) (char *filename, [ins_t](#) *instructions)
- int [countLines](#) (FILE *fp)

5.8.1. Documentación de las funciones

5.8.1.1. int countLines (FILE * fp)

5.8.1.2. void decodeInstruction (instruction_t instruction, uint32_t * Registro, char * R_Banderas)

Decodifica la instrucción y la ejecuta.

Parámetros

<i>instruction</i>	instrucción a decodificar y ejecutar
<i>*Registro</i>	Puntero al registro
<i>*R_Banderas</i>	Puntero a las banderas

5.8.1.3. instruction_t getInstruction (char * instStr)

Obtiene la instrucción separada por partes.

Parámetros

<i>instStr</i>	cadena que contiene la instrucción
----------------	------------------------------------

Devuelve

[instruction_t](#) la instrucción separada por partes

5.8.1.4. int readFile (char * filename, ins_t * instructions)

5.9. Referencia del Archivo desplazamiento.c

```
#include <stdio.h>
```

```
#include <stdlib.h>
#include <stdint.h>
#include "desplazamiento.h"
#include "ALU.h"
```

'defines'

- #define PC 15
- #define C 2

Funciones

- void **LSL** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion LSL Desplazamiento logico a la izquierda
- void **LSR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion LSR Desplazamiento logico a la derecha
- void **ROR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ROR Para el rotamiento a la derecha
- void **ASR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ASR Desplazamiento aritmetico a la derecha
- void **BIC** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)
Funcion BIC La cual niega bit a bit un registro y los multiplica
- void **MVN** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, char *R_Banderas)
Funcion MUN Niega un para metro bit a bit y lo guarda en otro
- void **RSB** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, int inmediato, char *R_Banderas)
Funcion RSB Niega un parametro
- void **NOP** (uint32_t *Registro)
Funcion NOP No hace nada en un tiempo o flanco de reloj
- void **REV** (uint32_t *Registro, uint32_t *Rd)
Funcion REV realiza grupos de 8 bits y los rota
- void **REVIG** (uint32_t *Registro, uint32_t *Rd)
Funcion REVIG realiza grupos de 16 bits y los rota
- void **REVSH** (uint32_t *Registro, uint32_t *Rd)
Funcion REVSH realiza extension de signo

5.9.1. Documentación de los 'defines'

5.9.1.1. #define C 2

5.9.1.2. #define PC 15

5.9.2. Documentación de las funciones

5.9.2.1. void ASR (uint32_t * Registro, uint32_t * Rd, uint32_t Rn, uint32_t Rm, char * R_Banderas)

Funcion ASR Desplazamiento aritmetico a la derecha

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundoegundo registro
<i>Rm</i>	Tercer registro o valor inmediato
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.9.2.2. void BIC (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rm*, char * *R_Banderas*)

Funcion BIC La cual niega bit a bit un registro y los multiplica

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	Primer registro
<i>Rm</i>	segundo registro
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.9.2.3. void LSL (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion LSL Desplazamiento logico a la izquierda

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundoegundo registro
<i>Rm</i>	Tercer registro o valor inmediato
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.9.2.4. void LSR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion LSR Desplazamiento logico a la derecha

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundoegundo registro
<i>Rm</i>	Tercer registro o valor inmediato
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.9.2.5. void MVN (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rm*, char * *R_Banderas*)

Funcion MUN Niega un para metro bit a bit y lo guarda en otro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	Primer registro
<i>Rm</i>	Segundo registro
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.9.2.6. void NOP (uint32_t * *Registro*)

Funcion NOP No hace nada en un tiempo o flanco de reloj

Parámetros

<i>*Registro</i>	Puntero al registro
------------------	---------------------

5.9.2.7. void REV (uint32_t * *Registro*, uint32_t * *Rd*)

Funcion REV realiza grupos de 8 bits y los rota

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	Primer registro

5.9.2.8. void REVIG (uint32_t * *Registro*, uint32_t * *Rd*)

Funcion REVIG realiza grupos de 16 bits y los rota

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro

5.9.2.9. void REVSH (uint32_t * *Registro*, uint32_t * *Rd*)

Funcion REVSH realiza extension de signo

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro

5.9.2.10. void ROR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ROR Para el rotamiento a la derecha

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundo registro
<i>Rm</i>	Tercer registro o valor inmediato
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.9.2.11. void RSB (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, int *inmediato*, char * *R_Banderas*)

Funcion RSB Niega un parametro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	Primer registro
<i>Rn</i>	segundo registro o valor inmediato
<i>inmediato</i>	para dar la condicion y negar el registro
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.10. Referencia del Archivo desplazamiento.h

```
#include <stdint.h>
```

Funciones

- void **LSL** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion LSL Desplazamiento logico a la izquierda
- void **LSR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion LSR Desplazamiento logico a la derecha
- void **ROR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ROR Para el rotamiento a la derecha
- void **ASR** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, uint32_t Rm, char *R_Banderas)
Funcion ASR Desplazamiento aritmetico a la derecha
- void **BIC** (uint32_t *Registro, uint32_t *Rd, uint32_t Rm, char *R_Banderas)
Funcion BIC La cual niega bit a bit un registro y los multiplica
- void **MVN** (uint32_t *Registro, uint32_t *Rd, uint32_t Rm, char *R_Banderas)
Funcion MUN Niega un para metro bit a bit y lo guarda en otro
- void **RSB** (uint32_t *Registro, uint32_t *Rd, uint32_t Rn, int inmediato, char *R_Banderas)
Funcion RSB Niega un parametro
- void **NOP** (uint32_t *Registro)
Funcion NOP No hace nada en un tiempo o flanco de reloj
- void **REV** (uint32_t *Registro, uint32_t *Rd)
Funcion REV realiza grupos de 8 bits y los rota
- void **REVIG** (uint32_t *Registro, uint32_t *Rd)
Funcion REVIG realiza grupos de 16 bits y los rota
- void **REVSH** (uint32_t *Registro, uint32_t *Rd)
Funcion REVSH realiza extension de signo

5.10.1. Documentación de las funciones

5.10.1.1. void ASR (uint32_t * Registro, uint32_t * Rd, uint32_t Rn, uint32_t Rm, char * R_Banderas)

Funcion ASR Desplazamiento aritmetico a la derecha

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundo registro
<i>Rm</i>	Tercer registro o valor inmediato
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.10.1.2. void BIC (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, char * *R_Banderas*)

Funcion BIC La cual niega bit a bit un registro y los multiplica

Parámetros

* <i>Registro</i>	Puntero al registro
* <i>Rd</i>	Primer registro
<i>Rm</i>	segundo registro
* <i>R_Banderas</i>	Puntero al registro de las banderas

5.10.1.3. void LSL (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion LSL Desplazamiento logico a la izquierda

Parámetros

* <i>Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundoegundo registro
<i>Rm</i>	Tercer registro o valor inmediato
* <i>R_Banderas</i>	Puntero al registro de las banderas

5.10.1.4. void LSR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion LSR Desplazamiento logico a la derecha

Parámetros

* <i>Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundoegundo registro
<i>Rm</i>	Tercer registro o valor inmediato
* <i>R_Banderas</i>	Puntero al registro de las banderas

5.10.1.5. void MVN (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, char * *R_Banderas*)

Funcion MUN Niega un para metro bit a bit y lo guarda en otro

Parámetros

* <i>Registro</i>	Puntero al registro
* <i>Rd</i>	Primer registro
<i>Rm</i>	Segundo registro
* <i>R_Banderas</i>	Puntero al registro de las banderas

5.10.1.6. void NOP (uint32_t * *Registro*)

Funcion NOP No hace nada en un tiempo o flanco de reloj

Parámetros

<i>*Registro</i>	Puntero al registro
------------------	---------------------

5.10.1.7. void REV (uint32_t * *Registro*, uint32_t * *Rd*)

Funcion REV realiza grupos de 8 bits y los rota

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	Primer registro

5.10.1.8. void REVIG (uint32_t * *Registro*, uint32_t * *Rd*)

Funcion REVIG realiza grupos de 16 bits y los rota

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro

5.10.1.9. void REVSH (uint32_t * *Registro*, uint32_t * *Rd*)

Funcion REVSH realiza extension de signo

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro

5.10.1.10. void ROR (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, char * *R_Banderas*)

Funcion ROR Para el rotamiento a la derecha

Parámetros

<i>*Registro</i>	Puntero al registro
<i>Rd</i>	Primer registro
<i>Rn</i>	Segundoegundo registro
<i>Rm</i>	Tercer registro o valor inmediato
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.10.1.11. void RSB (uint32_t * *Registro*, uint32_t * *Rd*, uint32_t *Rn*, int *inmediato*, char * *R_Banderas*)

Funcion RSB Niega un parametro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*Rd</i>	Primer registro
<i>Rn</i>	segundo registro o valor inmediato
<i>inmediato</i>	para dar la condicion y negar el registro
<i>*R_Banderas</i>	Puntero al registro de las banderas

5.11. Referencia del Archivo main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "mostrar.h"
#include "ALU.h"
#include "curses.h"
#include "desplazamiento.h"
#include "branch.h"
#include "decoder.h"
```

'defines'

- #define PC 15

Funciones

- int main ()

5.11.1. Documentación de los 'defines'

5.11.1.1. #define PC 15

5.11.2. Documentación de las funciones

5.11.2.1. int main ()

5.12. Referencia del Archivo mostrar.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "curses.h"
#include "mostrar.h"
#include "ALU.h"
```

'defines'

- #define N 0
- #define Z 1

- #define C 2
- #define V 3
- #define LR 14
- #define PC 15

Funciones

- void `mostrar_valores` (uint32_t *Registro, char *R_Banderas)
funcion para mostrar los datos del registro

5.12.1. Documentación de los 'defines'

5.12.1.1. #define C 2

5.12.1.2. #define LR 14

5.12.1.3. #define N 0

5.12.1.4. #define PC 15

5.12.1.5. #define V 3

5.12.1.6. #define Z 1

5.12.2. Documentación de las funciones

5.12.2.1. void `mostrar_valores` (uint32_t * *Registro*, char * *R_Banderas*)

funcion para mostrar los datos del registro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*R_Banderas</i>	Puntero a las banderas

5.13. Referencia del Archivo mostrar.h

```
#include <stdint.h>
```

Funciones

- void `mostrar_valores` (uint32_t *Registro, char *R_Banderas)
funcion para mostrar los datos del registro

5.13.1. Documentación de las funciones

5.13.1.1. void `mostrar_valores` (uint32_t * *Registro*, char * *R_Banderas*)

funcion para mostrar los datos del registro

Parámetros

<i>*Registro</i>	Puntero al registro
<i>*R_Banderas</i>	Puntero a las banderas

5.14. Referencia del Archivo README.md

5.15. Referencia del Archivo test.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <curses.h>
#include "mostrar.h"
#include "ALU.h"
#include "desplazamiento.h"
#include "branch.h"
```

'defines'

- #define N 0
- #define Z 1
- #define C 2
- #define V 3
- #define LR 14
- #define PC 15

Funciones

- int main ()

5.15.1. Documentación de los 'defines'

5.15.1.1. #define C 2

5.15.1.2. #define LR 14

5.15.1.3. #define N 0

5.15.1.4. #define PC 15

5.15.1.5. #define V 3

5.15.1.6. #define Z 1

5.15.2. Documentación de las funciones

5.15.2.1. int main ()