PSoC Instruction Set

Instructions	Flags	Cycles	Bytes	Opcode
adc A,expr	c,z	4	2	09
adc A,[expr]	c,z	6	2	0A
adc A,[X+expr]	c,z	7	2	0B
adc [exp],A	c,z	7	2	0C
adc [X+expr],A	c,z	8	2	0D
adc [expr],expr	c,z	9	3	0E
adc [X+expr],expr	c,z	10	3	0F
add A,expr	c,z	4	2	01
add A,[expr]	c,z	6	2	02
add A,[X+expr]	c,z	7	2	03
add [expr],A	c,z	7	2	04
add [X+expr],A	c,z	8	2	05
add [expr],expr	c,z	9	3	06
add [X+expr],expr	c,z	10	3	0F
add SP,expr		5	2	38
and A,expr	z	4	2	21
and A,[expr]	z	6	2	22
and A,[X+expr]	z	7	2	23
and [exp],A	z	7	2	24
and [X+expr],A	z	8	2	25
and [expr],expr	z	9	3	26
and [X+expr],expr	z	10	3	27
and F,expr	c,z	4	2	70
and reg[expr],expr	z	9	3	41
and reg[X+expr],expr	z	10	3	42
asl A	c,z	4	1	64
asl [expr]	c,z	7	2	65
asl [X+expr]	c,z	8	2	66
asr A	c,z	4	1	67
asr [expr]	c,z	7	2	68
asr [X+expr]	c,z	8	2	69
call		11	2	9x
cmp A,expr	c,z	5	2	39
cmp A,[expr]	c,z	7	3	3A
cmp A,[X+expr]	c,z	8	2	3B
cmp [expr],expr	c,z	8	3	3C
cmp [X+expr],expr	c,z	9	3	3D

Instructions	Flags	Cycles	Bytes	Opcode
cpl A	z	4	1	73
dec A	c,z	4	1	78
dec X	c,z	4	1	79
dec [expr]	c,z	7	2	7A
dec [X+expr]	c,z	8	2	7B
inc A	c,z	4	1	74
inc X	c,z	4	1	75
inc [expr]	c,z	7	2	76
inc [X+expr]	c,z	8	2	77
index	z	13	2	Fx
jacc		7	2	Ex
jc		5	2	Cx
jmp		5	2	8x
jnc		5	2	Dx
jnz		5	2	Bx
jz		5	2	Ax
lcall		13	3	7C
ljmp		7	3	7D
mov X,SP		4	1	4F
mov A,expr	z	4	2	50
mov A,[expr]	z	5	2	51
mov A,[X+expr]	z	6	2	52
mov [expr],A		5	2	53
mov [X+expr],A		6	2	54
mov [expr],expr		8	3	55
mov [X+expr],expr		9	3	56
mov X,expr		4	2	57
mov X,[expr]		6	2	58
mov X,[X+expr]		7	2	59
mov [expr],X		5	2	5A
mov A,X	z	4	1	5B
mov X,A		4	1	5C
mov A,reg[expr]	z	6	2	5D
mov A,reg[X+expr]	z	7	2	5E
mov [expr],[expr]		10	3	5F
mov reg[expr],A		5	2	60
mov reg[X+expr],A		6	2	61
mov reg[expr],expr		8	3	62
mov reg[X+expr],expr		9	3	63

Instructions	Flags	Cycles	Bytes	Opcode
mvi A,[[expr]++]	z	10	2	3E
mvi [[expr]++],A		10	2	3F
nop		4	1	40
or A,expr	z	4	2	29
or A,[expr]	z	6	2	2A
or A,[X+expr]	z	7	2	2B
or [expr],A	z	7	2	2C
or [X+expr],A	z	8	2	2D
or [expr],expr	z	9	3	2E
or [X+expr],expr	z	10	3	2F
or reg[expr],expr	z	9	3	43
or reg[X+expr],expr	z	10	3	44
or F,expr	c,z	4	2	71
pop X		5	1	20
pop A	z	5	1	18
push X		4	1	10
push A		4	1	8
reti	c,z	10	1	7e
ret		8	1	7F
rlc A	c,z	4	1	6A
rlc [expr]	c,z	7	2	6B
rlc [X+expr]	c,z	8	2	6C
romx	z	11	1	28
rrc A	c,z	4	1	6D
rrc [expr]	c,z	7	2	6E
rrc [X+expr]	c,z	8	2	6F
sbb A,expr	c,z	4	2	19
sbb A,[expr]	c,z	6	2	1A
sbb A,[X+expr]	c.z	7	2	1B
sbb [expr],A	c,z	7	2	1C
sbb [X+expr],A	c,z	8	2	1D
sbb [expr],expr	c,z	9	3	1E
sbb [X+expr],expr	c,z	10	3	1F
ssc	c,z	15	1	00
sub A,expr	c,z	4	2	11
sub A,[expr]	c,z	6	2	12
sub A,[X+expr]	c,z	7	2	13
sub [expr],A	c,z	7	2	14
sub[X+expr],A	c,z	8	2	15

Instructions	Flags	Cycles	Bytes	Opcode
sub [expr],expr	c,z	9	3	16
sub [X+expr],expr	c,z	10	3	17
swap A,X	z	5	1	4B
swap A,[expr]	z	7	2	4C
swap X,[expr]		7	2	5D
swap A,SP	z	5	1	5E
tst [expr],expr	z	8	3	47
tst [X+expr],expr	z	9	3	48
tst reg[expr],expr	z	9	3	49
tst reg[X+expr],expr	z	10	3	4A
xor F,expr	c,z	4	2	72
xor A,expr	z	4	2	31
xor A,[expr]	z	6	2	32
xor A,[X+expr]	z	7	2	33
xor [expr],A	z	7	2	34
xor [X+expr],A	z	8	2	35
xor [expr],expr	z	9	3	36
xor [X+expr],expr	z	10	3	37
xor reg[expr],expr	z	9	3	45
xor reg[X+expr],expr	z	10	3	46

Assembler Directives	
area	area
Null Terminates String	asizz
Reserve RAM bytes	blk
Reserve RAM words	blkw
Define Byte	db
Define String	ds
Define Unicode String	dsu
Define Word	dw
Define Word little endian	dwl
Equate	equ
Export	export
IF, ELSE, ENDIF	IF, ELSE, ENDIF
Include	include
Prevent Compression	.literal, .endliteral
Macro Definition	macro, endm
Area Origin	.org
Sections	.section .endsection

Register Definition	Bank	Name
Port x Data Register	0	PRTxDR
Port x Interrupt Enable	0	PRTxIE
Port x Global Select	0	PRTxGS
Port x Drive Mode 2	0	PRTxDM2
Port x Drive Mode 0	1	PRTx0DM0
Port x Drive Mode 1	1	PRTx0DM1
Port x Interrupt Control 0	1	PRTx0IC0
Port x Interrupt Control	1	PRTx0IC1
Analog Input Multiplexer	1	AMX_IN
I2C Configuration Register	0	I2C_CFG
I2C Status and Control	0	I2C_SCR
I2C Data Register	0	I2C_DR
I2C Master Status and Control	0	I2C_MSCR
I2C and Software Mask	0	INT_MSK3
General Interrupt Mask	0	INT_MSK0
Digital PSoC block Mask	0	INT_MSK1
Decimator Register	0	DEC_DH
Decimator Register	0	DEC_DL
Decimator Control Register	0	DEC_CR0
Decimator Control Register	0	DEC_CR1
Multiplier X Register	0	MUL_X
Multiplier Y Register	0	MUL_Y
Multiplier Result Data	0	MUL_DH
Multiplier Result Data)	0	MUL_DL
MAC X register	0	MAC_X
MAC Y register	0	MAC_Y
MAC Clear Accum]	0	MAC_CL0
MAC Clear Accum	0	MAC_CL1
MAC Result	0	ACC_DR1
MAC Result	0	ACC_DR0
MAC Result	0	ACC_DR3
MAC Result	0	ACC_DR2
Analog Modulator Control	0	AMD_CR0
Analog Modulator Control	0	AMD_CR1

Assemby Syntax	Symbol	Form
Bitwise Compliment	~	(~a)
Mult/Div/Mod	*,/,%	(a*b),(a/b),(a%b)
Add/Sub	+,-	(a+b),(a-b)
Bitwise AND	&	(a&b)
Bitwise XOR	^	(a^b)
Bitwise OR	|	(a|b)
High Byte	>	(>a)
Low Byte	<	(<a)

Fastcall Argument Passing		
Argument Type	Register	
char	A	
char,char	A,X	
int	X,A	256*X+A
pointer	A,X	256*A+X
Fastcall Argument Return		
Return Type	Register	
char	A	
int	X,A	256*X+A
pointer	A,X	256*A+X

Usefull Macros	
M8C_EnableGInt	M8C_Stall,
M8C_DisableGInt	M8C_Unstall
M8C_EnableWatchDog	M8C_SetBank0
M8C_ClearWDT	M8C_SetBank1
M8C_ClearWDTAndSleep	M8C_ClearIntFlag
M8C_DisableIntMask	M8C_Sleep
M8C_EnableIntMask	M8C_Reset

Apps. Hotline **1 800.669.0557 x4814**

CYPRESS MICROSYSTEMS