Tutorial Microcontrolador PIC (versión 3.5)

En este curso básico de microntroladores PIC se estudiará el PIC 16f84, por ser este el de uso más común entre los estudiantes que se inician en el tema.

El primer paso importante el ver el diagrama de pines del PIC16F84, en el cual se observa como están distribuidos sus pines.

Este circuito integrado cuenta con 2 puertos configurables como entradas o salidas según sea el caso y consta de 18 pines las cuales se encuentran asignadas de la siguiente manera:

VDD: Tensión positiva de alimentación.

VSS: Tensión conectada a tierra o negativa de alimentación. O sea entre los pines 14 y 5 VDD(+)VSS(-) se coloca la alimentación la cual no debe sobrepasar los 5 Voltios.

OSC1/CLKIN: Entrada del circuito oscilador externo que proporciona la frecuencia de trabajo del microcontrolador.

OSC2/CLKOUT: Patilla auxiliar del circuito oscilador.

MCLR#: Patilla activa con nivel lógico bajo, lo que se representa con el símbolo # o con una línea superior MCLR. Su activación origina la reinicialización o Reset del PIC. El pin 4 (MCLR#), o sea, el Reset se debe conectar con una resistencia de 10 Kohm a Vcc para que el Pic funcione, si lo queremos resetear entonces pondremos un micropulsador con una resistencia de 100 Ohm a tierra. También se usa durante la grabación de la memoria para introducir por ella la tensión VPP.

RAO-RA4: Son las 5 líneas de E/S digitales correspondientes a la Puerta A. La línea RA4 multiplexa otra función expresada por TOCKI. En ese caso sirve para recibir una frecuencia externa para alimentar al temporizador TMRO. O sea, RA4/TOCKI puede ser configurado a su vez como entrada/salida o como temporizador/contador. Cuando es salida se comporta como colector abierto, por lo tanto debemos poner una resistencia Pull-up a Vcc de 1 Kohm. Cuando es configurada como entrada, funciona como disparador Schmitt Trigger por lo que puede reconocer señales con un poco de distorsión.

RBO-RB7: Son las 8 líneas de E/S digitales de la Puerta B. La línea RBO multiplexa la función de servir como entrada a una petición externa de una interrupción.

La máxima	capacidad de corriente					
para cada	uno de los pines de los					
puertos se muestra en la tabla						

te		PUERTO A	PUERTO B
los	MODO SUMIDERO (sink)	25mA	25 mA
	MODO FUENTE (source)	20 mA	20 mA

Oscilador Externo:

Es necesario para que nuestro PIC pueda funcionar, puede ser conectado de cuatro maneras diferentes. Se muestran en la siguiente tabla.

XT	Oscilador compuesto por un cristal y dos condensadores	Z2pF ra2 ra1 ra3 ra0 ra4oscl rstosc2 -rb0 rb7 rb1 rb6 rb2 rb5 -rb3 rb4 -rb3 rb4
RC	Oscilador compuesto por una resistencia y un condensador.	PIC16F84 ra2 ra1 ra3 ra0 rstosc2 cmd vb7 rb0 rb7 rb1 rb5 rb3 rb4



Prof: Bolaños D. Electrónica

HS	Oscilador compuesto por un cristal de alta velocidad.	Con cristal.
LP	Oscilador compuesto por un cristal de baja frecuencia y bajo consumo de potencia.	Con cristal.

El siguiente paso importante para tener claro como debemos empezar a programar es conocer la tabla de registros. Esta tabla está dividida en dos partes llamadas BANCO O y Banco 1. Los registros más importantes para comenzar son: STATUS, PORTA, PORTB, TRISA y TRISB.

Para que nuestro PIC pueda trabajar debemos configurar sus puertos como entrada o como salida según sea el caso.

Esta asignación de pines de puertos como entrada o como salida se hace programando los registros TRISA y TRIS B.

TRISA es el registro donde se almacenan los bits que asignan un pin como entrada o salida del PUERTO A. Recordemos que el puerto A sólo tiene 5 pines, por lo tanto un ejemplo de esto sería:

Si TRISA (puerto A) es igual a 19h (en HEXA) o (0011001 en binario) entonces esto se leería,

TRISA	ASIGNACIÓN	ESTADO
RA0	1	ENTRADA
RA1	0	SALIDA
RA2	0	SALIDA
RA3	1	ENTRADA
RA4	1	ENTRADA

El bit menos significativo sé asigna desde RAO.

Si TRISB (puerto B) es igual a 32h (en HEXA) o (00110010 en binario), entonces esto se leería,

TRISB	ASIGNACIÓN	ESTADO
RB0	0	SALIDA
RB1	1	ENTRADA
RB2	0	SALIDA
RB3	0	SALIDA
RB4	1	ENTRADA
RB5	1	ENTRADA
RB6	0	SALIDA
RB7	0	SALIDA

NOTA: La asignación de valor a un registro se puede hacer en HEXA (ej:0x19), o en DECIMAL (Ej:d'12'), o en BINARIO (Ej:b'00101110').

A continuación comenzaremos a programar el PIC y veremos como ingresar estos valores en el TRIS A o TRIS B según sea el caso.

Para trabajar con los microcontroladores se debe conocer y manejar alguna herramienta de desarrollo. MPLAB es la herramienta de desarrollo de microcontroladores PIC. (Microchip, Inc., los creadores de los microcontroladores PIC).

Si no estas familiarizado con las herramientas de desarrollo avanzadas, quizás encuentres el MPLAB un poco confuso al principio. También se necesita conocer el lenguaje de programación PIC.

En vez del MPLAB, se puedes usar el MPASM por ejemplo, un programa basado en MS-DOS, pero en esta explicación utilizaremos el MPLAB.

Todo en el MPLAB gira en torno al concepto de "project" (proyecto), que es un conjunto de archivos que necesitan ser procesados para compilar tu programa.

Para comenzar a trabajar con el programa MPLAB debemos simplemente abrir un nuevo archivo en la ventana File opción New como muestra la siguiente figura:

	MPLAB										
File	Project	Edit	Debug	Picstart Plus	Options	Tools	Window	Help			
- N	lew			Cl	:rl+N				F	TOT RATE S	गा लगा इस 📼
	Open			Ct	rl+0			<u> </u>			<u>لللل</u> المتعاركة
۷	/iew										
S	iave			C	trl+S						
S	iave As										
S	iave All										
0	lose			-1.5							
	lose All			Shif	t+F9						
I	mport				F						
E	Export				•						
P	rint			C	trl+P						
P	rint Setup										
E	Exit			Al	t+F4						
1	c:\archiv	√2\ele	ctronimis	progitarea4b./	asm						
2	c:\archiv	∿2\ele	ctron'mis	proglaioa ibi. proglaisplav4	asm						
3	: c:\archiv	~2\ele	ctron'mis	proglaspia, in proglareas4.2	asm						
4	c:\archiv	~2\ele	ctron'mis	progleancion1.	asm						
5	i c:\archiv	∼2\ele	ctron\mis	progleandon i progleandos.a	sm						
	, enjarenni	-1010		.p. og (501 lides i e							

Iniciando un nuevo programa.

A continuación se lista un ejemplo de programa:

	list	p=16f8	4					
	#inclu	ide p16f	84.inc					
porta	equ 05	5h						
- trisa	equ 85	5h						
status	equ 03h							
mio	equ Oc	ch						
inicio			;comienzo del programa					
	movlw	0xaa	;carga 10101010 en w					
	clrw		;limpia w					
	clrf	mio	;limpia registro mio					
	movlw	0xf0	; carga 11110000 en w					
	movwf	mio	;mueve w a mio					
	movlw	0x00	;carga 00000000 en w					
	bsf	status	,5 ;se ubica en el bancol de la ram					
	movwf	trisa	;carga w en trisa asi define porta como salida					
	bcf	status	,5					
	clrf	porta	;limpia reg porta					
	movlw	0x0f	;carga 00001111 en w					
	movwf	porta	; mueve w en porta					
	clrw							
	movlw	0x0a	;carga 00001010 en w					
	addlw	B'101'	;adiciona literal binario 101 a w					
	movwf	mio	; mueve w en mio					
	comf	mio,1	;complementa mio y guarda en mio por ser d=1					
	inci	mio,1	;incrementa mio y guarda en mio por ser d=1					
	swapı	mio,i	;Intercampia nibles de mio y guarda en mio por ser d=1					
	movi	m10,0	;mueve mio en w porque d=0					
	deci	mio,i	Jaerementa mio y guarda en mio por ser d=1					
	deci	mio,I	; loem anterior					
	olmu	m10,0	prueba si bit 0 de mio es cero. Si 10 es salta la proxima instrucción					
	gata	inicio	valta al programa a inicia					
	golo	TUTCTO	, sarca er programa a inicio unare informente el commileder que termine el listado del programe					
	ena		,para informatie al compliador que termina el listado del programa					

RECOMENDACIÓN: Siempre tener a mano el set de instrucciones.

Debemos destacar lo siguiente:

✓ Para mantener un orden, se recomienda presionar la tecla TAB entre la instrucción y el parámetro de la instrucción (Llamaremos parámetro a aquello que acompaña a la instrucción para que la misma se lleve a cabo). ✓ El símbolo de Punto y Coma " ; " indica el comienzo de un comentario, sirve para ubicarse mejor en el avance del programa y luego encontrar mas rápidamente una línea o bloque determinado. Todo lo que siga a un " ; " no generará código en nuestro proyecto. Lo que siga al " ; " no influye en el programa.

Analicemos las líneas de programa ejemplo:

	list p=16f84 #include p16f84.inc	Primero debemos especificar con que microcontrolador estamos trabajando, en las dos primeras líneas especificamos que vamos a trabajar con el PIC 16F84.
porta trisa status mio	equ 05h equ 85h equ 03h equ 0ch	Luego definimos la posición en memoria de nuestros registros, el mapa de memoria RAM se presenta a continuación

Para comenzar destacaremos los registros:

porta	posición	05h
portb	posición	06h
trisa	posición	85h
trisb	posición	86h
status	posición	03h
mio	posición	0ch

Definimos los nombres de nuestros registros, en el caso del Puerto A lo llamamos porta, es importante que siempre que nos queramos referir al Puerto A lo hagamos con el nombre asignado. Por ejemplo si en lugar de "porta equ 05h" colocamos "hola equ 05h" cada vez que queramos referirnos al Puerto A debemos hacerlo como "hola". También se debe respetar si lo escribimos en mayúsculas o minúsculas. Mi recomendación es respetar los nombres de la figura del mapa de la RAM.

Puede suceder que cuando vemos el listado de un programa, omitan el listado de registros especiales (o sea los que aparecen en el mapa de la RAM), esto es debido a que la instrucción **#include pl6f84.inc** hace el llamado a un archivo librería que contiene este listado. Mi recomendación es colocar el listado de los registros especiales que usaremos en el programa para así mejorar la lectura y revisión del mismo.

El registro **mio** es un registro de propósito general, en este caso localizado en la posición de memoria **Och**, los nombres de estos registros los elegimos nosotros.

		_	
File Addre	ss Indiroct oddr (1)	F Indiroct addr (1)	lle Address
0011			0011
01h	TMR0	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h			87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2 ⁽¹⁾	89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	68 General Purpose registers (SRAM)	Mapped (accesses) in Bank 0	8Ch
4Fh 50h			CFh D0h
7Fh			FFh
	Bank 0	Bank 1	
🗌 Unimpl	emented data mer	nory location; read	as '0'.
Note 1:	Not a physical reg	ister.	

Mapa de la memoria RAM

En algún momento deberemos estudiar la función de cada bit de los registros especiales, por ejemplo el registro STATUS se compone de la siguiente forma:

7	6	5	4	3	2	1	0
IRP	RP1	RP0	TO	PD	Ζ	DC	С

Pero aquí mencionaremos solo la función de dos de sus bits

✓ Si en el bit 5 (RPO) del registro STATUS hay un CERO entonces estamos en el BANCO 0.
 ✓ Si en el bit 5 (RPO) del registro STATUS hay un UNO entonces estamos en el BANCO 1.
 Anteriormente ya mencionamos la función de los registros trisa y trisb.

Al lado del resto de las instrucciones aparece una breve descripción de su función.

Finalmente debemos guardar este programa, para esto vamos al menú **File**, luego **Save As** y elegimos el directorio y un nombre, en este caso le pondremos <u>ejemplo.asm</u>, luego **OK**.

MPLAB	
Hie Project Edit Debug Picstart Plus Options Tools window Heip	
Intitled1	
list p=16f84	
porta Save File As 🗙	
trisa status File Name: Directories:	
mio ejemplo.asm c:\archiv~2\electron\pic	
Cancel	
BAJALO~1	
SETDEL"1 V Keen backup	
List Files of Type: Drives: Red	
Source Files (c,h,asm,ir 🔽 🗏 C:	104
- ozi i por ca jizmpia i cy por ca	
movlw 0x0f ;carga 00001111 en w	
movw r porta ;mueve w en porta	

Veremos a ahora como compilar este programa y detectar los errores de programación.

Lo primero es hacer clik en el menú **Project** y luego en **New project**, como indica la figura:

	MPLAB								
File	Project	Edit	Debug	Picstart Plus	Options	Tools	Window	Help	
	New	Project	t			- b			801 FM 57 🔤 🍸
8	Close	: Projec : Projec	ct		Ctrl+F2				
lis	Save	Projec	:t						
DOP	Edit F	roject			Ctrl+F3				
tri	Make	Projec	t		F10				
sta	Build	All			Ctrl+F10				
M10	Build	Node			Alt+F10				
ini	Insta	ll Lang	uage Too	l		am	а		
1100	1 c:\archiv~2\electron\misprog\tarea4b.pjt 2 c:\archiv~2\electron\misprog\display4.pjt 3 c:\archiv~2\electron\misprog\tarea4.pjt 4 c:\archiv~2\electron\misprog\luces1.pjt 5 c:\archiv~2\electron\misprog\cancion1.pjt						ia regi 000 en e w a r 00 en v ubica e	istro w nio v en el	o mio L banco1 de la ram
	Γ	novwł	F tr	isa ;c	arga w	en t	risa ag	si de	efine porta como salida
	C	:lrf	po	rta ;]	impia r	eg p	orta		00 H
	Γ	novwł	F pa	rta ;	iueve w	en p	a oooo orta		en w
•									

El siguiente paso es darle un nombre al proyecto, se recomienda darle el mismo nombre que al archivo en assembler, en este caso <u>ejemplo.pjt</u>.



Una vez echo esto, aparece la siguiente ventana:



En la parte de abajo, donde dice **Project Files** pinchamos la opción <u>ejemplo [.hex]</u> al hacer esto se activa la opción **Node Properties** que esta abajo a la derecha, debemos pinchar esta y se abre la siguiente ventana:

📰 MPLA	B						
File Projec	t Edit Debu	g Picstart Plus Options Too	ols Window He	elp			
B				BOT RAT	97 👼 📅		
C:\ar	chiv-2\ele						- Interio
lict	n=16£84	lode Properties					
porta trisa status	p 10104	Node: EJEMF	PLO.HEX	<u> </u>	Language Tool:	MPASM	•
NITO		Description					Data
inicio		Define	💷 On	-			
MOVIN	UXaa clew	Hex Format	MINHX8M	INHX8S	INHX32		
	movlw movlw movwf clrf	Error File	🗾 On	🔳 Off			
		List File	🗾 On	🔳 Off			
		Cross-reference File	🔳 On	🗾 Off			
		Warning level	🗾 all	🔳 warn+err	err 📃		
		Case sensitivity	🗾 On	Off			
		Macro expansion	🗹 On	Off			
		Default radix	MEX HEX	DEC	🔳 ОСТ		
	movwf	Tab size	💷 On				
•							
		Command Line /alNHX8M /e+ /l+ /x- /u Additional Command Line	w0 /c+ /m+ /rł e Options	nex /p16F84			
			ОК		Cancel	Help	

En esta sección, debemos marcar como muestra la figura y luego OK. Luego volveremos a la ventana anterior solo que ahora se activó la opción Add Node algunas opciones más arriba de la última que pinchamos, por supuesto elegimos esta y se ve la siguiente pantalla:



Damos click en **Browse** que esta a la derecha y buscamos en el directorio el programa original, en este caso <u>ejemplo.asm</u> y luego en **OK**, con esto estamos relacionando un archivo de assembler con el proyecto en hexa.

Y finalmente volvemos a la primera pantalla y le damos al \mathbf{OK} que esta en la parte superior.

A continuación debemos verificar que no haya errores para eso vamos nuevamente a la opción **Project** y veremos que hay nuevas opciónes que antes no estaban disponibles, elegimos **Build All**, y si todo esta correcto aparecerá una pantalla como esta:

MPLAB - C:\ARCHIV~2\ELECTRON\PIC\EJEMPLO.PJT File Project Edit Debug Picstart Plus Options Tools Window Help X 🖻 🖻 日 Ê 🔛 🔍 ROM RAM SEB 🗵 c:\archiv~2\electron\pic\ejemplo.asm - 🗆 🗙 1i: Build Results - 🗆 🗙 po Building EJEMPLO.HEX... ۰ tr st.Compiling EJEMPLO.ASM: mi[,]Command line: "C:\ARCHIV~1\MPLAB\MPASMWIN.EXE /aINHX8M /e+ /l+ /x- /w0 /c+ /m+ /rh in Build completed successfully. mo

Donde en la parte final podemos leer: **Build completed successfully**. Esto implica que el program se compilo satisfactoriamente.

En caso de haber algún error, podemos hacer doble clic en el mensaje de error y este nos enviará directamente a esta línea.



Una vez que hemos pasado esta etapa, nos dedicamos a simular el programa, todo dentro del MPLAB.

Esto consiste en ver paso a paso que está realmente haciendo el PIC, ya que si lo hacemos en tiempo real es demasiado rápido. Claro que también se puede dar el caso contrario, es decir tener un programa con tiempos largos que tampoco podríamos simular porque tardaría demasiado, en ese caso la solución es alterar algunas líneas de programa que maneje los tiempos y luego proceder a simularlo.

Lo primero es activar la simulación, para esto vamos al menú **Options** y le elegimos la opción **Development Mode**.

En la ventana que aparece marcamos donde dice MPLAB-SIM Simulator y además buscamos el PIC que estamos usando, en este caso el PIC16F84.

ile Project Edi	E Debug Picstart Plus Options Tools	
🗐 c:\archiv~	2\electron\pic\ejemplo.asm	
lis orta risa	t p=16f84 #include p16f84.inc equ 05b Development Mode	×
tatus io	Development Modes:	Development Mode Options:
nicio mov clr mov mov bsf	 Editor <u>Only</u> MPLAB-SIM <u>Simulator</u> MPLAB-ICE Emulator PICMASTER <u>E</u>mulator SIM<u>I</u>CE Simulator ICEPIC MPLAB-ICD Debugger 	Processor: PIC16F84

Primero analizaremos las herramientas básicas para hacer una buena simulación, en la figura siguiente están marcados los principales botones para este trabajo.

MPLAB - C:\ARCHIV~2\ELECTRON\PI	PIC\EJEMPLO.PJT	
File Project Edit Debug Picstart Plus Optio	tions Tools Window Help	
) 🕈 🐿 🚬 🚛 🔟 🕅 🎛 📼 🍸	
1	2345578	

- 1. Conmutador de la barra de herramientas: Al pinchar esta venta tenemos mas opciones a nuestra disposición.
- (Run)Ejecuta la simulación en "tiempo real" (no olvidemos que en el simulador emula el funcionamiento del microcontrolador y es mucho más lento que este).
- 3. -Halt the processor, se detiene la ejecución del programa.
- 4. Step: Avanza paso a paso por las instrucciones del programa.

5. -

- 6. Reset: comienza de cero el programa, equivale a activar el pin N°4 del PIC, Master Clear (MCLR).
- 7. Special Function Register Window: Muestra el estado de los registros especiales, por ejemplo el registro status, puerto a y b, el acumulador, etc.
- 8. Watch Symbol: Ventana en la que podemos ver los registro de propósito general, que son los registro que usamos en nuestro programa y a los cuales de damos nombres propios.

Al presionar sobre el botón N°7 $({\bf FSR})\,$ aparece una ventana donde observamos los registros especiales.

Al presionar sobre el botón N°8 aparece otra ventana donde debemos elegir (**Add**) los registros de propósito general que queremos controlar, en **Properties** el formato de presentación (BINARIO - DECIMAL etc), al terminar hacemos click en **Close**.



Ahora, para comenzar la simulación es recomendable ordenar las ventanas de forma que entren todas en la pantalla.

MPLAB - C:\ARCHIV~2\ELECTRON\PIC\EJEMPLO.PJT					- 🗆 ×
File Project Edit Debug Picstart Plus Options Tools Window Help					
	9R 🏧 🍸				
📱 c:\archiv~2\electron\pic\ejemplo.asm 📃 🗖 🗙	🗵 Special	Function I	Register	Window	- 🗆 🗙
16f84_inc	SFR Name	Hex 88	Dec	Binary AAAAAAAAA	Char
	DC1	00	0	00000000	
	option	FF	255	11111111	.
	status	18	24	00011000	.
	fsr	00	0	00000000	.
	porta	00	6	00000000	· · · · · · · · · · · · · · · · · · ·
del programa	trisa	1F	31	00011111	· ·
arga 10101010 en w	portb	00	0	00000000	·
impia w	trisb	FF	255	11111111	· · /
o ;limpia registro mio	eedata	00	0	00000000	·
carga 11110000 en w	eecon1	00	0	00000000	·
o ;mueve w a mio	eeadr	00	0	00000000	I
arga 00000000 en w	eecon2	00	0	00000000	·
se ubica en el banco1 de la ram	pclath	00	0	00000000	·
arga w en trisa asi define porta como salida	intcon	00	0	00000000	·
	W	00	0	00000000	·
impia reg porta	tOpre	00	0	00000000	I
0f ;carga 00001111 en w					I
ueve w en porta					
arga 00001010 en w	🖺 Watch_	2			- 🗆 🗙
101' ;adiciona literal binario 101 a w	Address Su	ymbol Va	lue		
ueve w en mio	0C mi	o B'	000000	100'	I
o,1 ;complementa mio y quarda en mio por ser d=1					
ncrementa mio y quarda en mio por ser d=1					I
o,1 ;intercambía nibles de mio y quarda en mio por 📒					I
ueve mio en w porque d=0					I
o,1 ;decrementa mio y quarda en mio por ser d=1 📃					I
dem anterior					
Tutorial microontroladores PI	C Microsoft Inte	ernet Explore	er - [Traba	ajar sin c	

Luego sólo debemos pinchar el botón Step (N°4) sucesivamente y con eso se ejecutará y se marcará la línea del programa que esta trabajando, otra opción es presionar F7.

<u>Para simular</u>

las entradas de un PIC

Vamos al menú Debug -

Simulator Stimulus -

Asynchronous Stimulus.

	MPLAB	- C:V		2\ELECTRO	N/PIC/E	JEMPL	.0.PJT				
File	Project	Edit	Debug	Picstart Plus	Options	Tools	Windov	v Help			
-] 🛃	<u> </u>	Run				*		ROT RAT	SR 🔤	ľ
			Exec	ute			•			1	
	c:\arch	niv~2	Simul	ator Stimulus			Þ	Asyno	thronous Stir	nulus	ci
	נ	list	Cent	er Debug Loca	tion			Pin St Clock	imulus Stimulus	•	n
por	ta		Break	Settings			F2	Regis	ter Stimulus	•	L
tri	sa		Trace	e Settings						option	۳.
sta	itus		Triga	er In/Out Setti	ings			I		statu	5
mio	1		Clear	All Points	2			I		fsr	
ini	cio		Com	lev Trigger Se	ttings			1		trisa	
	п	novlu	Epob	la Cada Cauar	odnigorini			ω		portb	
	C	:lrw		ie Code Cover	aye					trisb	
			Clear	Program Mem	ory C	trl+Shift	:+F2	istro	nio 📃	eedata	a
	Γ	novlu	Syste	em Reset	C	trl+Shift	t+F3	ιw		eecon	1
			Powe	r-On-Reset	С	trl+Shift	t+F5	mio		eeadr	
	Г	novlu						W		eecon	2

Luego aparecerá una ventana en la cual debemos elegir que pin le vamos a asignar a cada opción.

Posicionados sobre el primero elegimos con el botón derecho del mouse la opción Assign Pin y luego elegimos el pin que le vamos a designar haciendo doble click sobre este. En la figura se va a elegir el RAO.

PLAB - C:\ARCH	IIV~2\ELECTRO	4/PIC/EJ	empl	0.PJT	
roject Edit Deb	ug Picstart Plus	Options -	Tools	Window Help	
	XBB] 👌			ROT R
Asynchronous	Stimulus Dialo				×□
, no jiiciii ono a		5			
RAO (P)	Stim 2 (P)	Stim 3	(P)	Stim 4 (P)	
Stim 5 (P)	Pin Selection	×	P)	Stim 8 (P)	
Stim 9 (P)	Double click t	o select.	(P)	Stim12 (P)	
10 movlw clrw movlw movlw bsf movwf	MCLR RA0 RA1 RA2 RA3 RA4 RB0 RB1 RB1 RB2 RB3 RB3 RB4	<u> </u>	rama 0101 1100 ueve 0000 en n tr	a 10 en w 300 en w 2 w a mio 30 en w el banco1 risa asi de) mio de la efine
	PLAB - C:VARCH roject Edit Debi File A Asynchronous RAO (P) Stim 5 (P) Stim 9 (P) 10 movlw clrw movlw bsf movlw	PLAB - C: VARCHIV - 2VELECTROP roject Edit Debug Picstart Plus PLAB - C: VARCHIV - 2VELECTROP roject Edit Debug Picstart Plus PLAB - C: VARCHIV - 2VELECTROP Picstart Plus PLAB - C: VARCHIV PLAB	PLAB - C: VARCHIV - 2VELECTRON/PICVEJ roject Edit Debug Picstart Plus Options PLAB - C: VARCHIV - 2VELECTRON/PICVEJ roject Edit Debug Picstart Plus Options PLAB - C: VARCHIV - 2VELECTRON/PICVEJ Plasma - Comparison of the select Asynchronous Stimulus Dialog RA0 (P) Stim 2 (P) Stim 3 Stim 5 (P) Pin Selection Stim 9 (P) Double click to select. MCLR RA0 movlw clrw RA1 RA2 RA3 movlw RA4 RB0 RB1 RB1 RB1 RB1 RB1 RB1 RB1 RB1	PLAB - C: VARCHIV - ZVELECTRONNPIC/EJEMPL roject Edit Debug Picstart Plus Options Tools Image: Start Plus Im	PLAB - C:VARCHIV - 2VELECTRONVPIC/EJEMPLO.PJT roject Edit Debug Picstart Plus Options Tools Window Help Image: Start Plus Image: Start Plus

Luego es muy importante decirle al MPLAB que vamos a hacer una simulación paso a paso y no en tiempo real, para esto una vez que le hemos designado el pin correspondiente, volvemos a hacer click con el botón secundario del mouse pero esta vez elegimos la opción Toggle. Después de esto podemos observar que ya no hay una (P) a la derecha de botón, sino que ahora hay una (T).

-1	Asynchronous S	timulus Dialo	g		×
	RAO (T)	RA1 (T)	Assian Pin	RA3 (P)	1
	RA4 (P)	Stim 6 (P)		Stim 8 (P)	
	Stim 9 (P)	Stim 10 (P	✓ Pulse Low	Stim12 (P)	
			High		
			Toggle		
			Help		

Esta será una ventana mas que quedará abierta en pantalla. Cada vez que demos click sobre RAO (T) cambiara su estado. Obviamente esto deberá hacerse durante la simulación.

RESUMIENDO:

Se debe seleccionar la ventana del programa para que al presionar F7 en el teclado o la botón Step en la parte superior, se produzca la simulación, es decir, para que corra el programa.

En la ventana de funciones especiales, se pone rojo él ultimo registro en cambiar. Al igual que en el punto anterior, en la ventana de registros de propósito general, se pone rojo el ultimo en cambiar. Si queremos agregar mas registros para poder verlos en esta ventana debemos presionar en la parte superior izquierda y elegir la opción **Add Watch**.

Una vez satisfechos con la simulación, se debe cargar el programa en el PIC. Para esto usaremos el ICPROG. Hay apunte sobre el tema.