#### UNIVERSIDAD DE ANTIOQUIA DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA CURSO DE SISTEMAS EMBEBIDOS

### UTILIZACIÓN DE CODEWARRIOR V.2.0 CON SIMULACIÓN DE PEMICRO Para proyectos en lenguaje c para microcontroladores HC08

Elaborado por : Jorge Andrés Baena Revisado por : Mauricio Álvarez Mesa Versión : 1.0 (11/06/2002) : <u>abaena@microe.udea.edu.co</u>

: <u>malvarez@udea.edu.co</u>

### Introducción

Metrowerks y Motorola han dejado disponible al público una edición especial del entorno de desarrollo para microcontroladores HC08: *CodeWarrior v2.0*, la cual es libre y puede compilar hasta 4Kbytes de código C. Una de las ventajas importantes de esta nueva versión es que adiciona el simulador de P&E Microcomputer Systems Inc., el cual cuenta con una máquina virtual que permite simular la CPU, periféricos e interrupciones de todos los microcontroladores HC08 actuales, lo que facilita el proceso de depuración de las aplicaciones desarrolladas en lenguaje C.

El instalador completo de CodeWarrior v2.0 se puede obtener por ftp (anónimo) desde el servidor del grupo de Microelectrónica: <u>ftp://microe.udea.edu.co</u> ingresando al directorio: **pub/embebidos/codewarrior** 

y descargando los archivos:

hc08codewarrior20.exe (Programa instalador) y license\_specialv2.0.zip (Archivo de licencia)

Para mayor información puede consultar directamente la página de Metrowerks: http://www.metrowerks.com/embedded/motoHC08/

O visitar periódicamente la página del curso de Sistemas Embebidos de la Universidad de Antioquia para nuevos tutoriales y ejemplos http://microe.udea.edu.co/cursos/ieo-944

### **Comenzando con proyectos**

Este tutorial es una introducción rápida para crear proyectos en lenguaje C, utilizando el compilador Metrowerks CodeWarrior v2.0 y el simulador ICS de PEMICRO. Para el tutorial se desarrolló un programa en C para el microcontrolador HC08GP32, el código fuente esta disponible en <u>http://microe.udea.edu.co/cursos/ieo-944/files/demoHC08</u>

El primer paso es crear el proyecto. Para esto seleccione File | New...

Con esto aparecerá la ventana que se muestra en la figura 1. Escoja **HC08 Stationery**, la carpeta donde va a ubicar su proyecto y el nombre del proyecto.

New Project File Object	×
Empty Project HC08 Stationery	Project name: example08 Location: D:\Usuarios\abaena\example0 Set Add to Project: Project: T
	Aceptar Cancelar

Figura 1. Crear proyecto nuevo

Cuando presione **Aceptar**, aparecerá otra ventana para escoger el tipo de sistema de desarrollo que va a utilizar (**Stationery**), como se desea utilizar la simulación de PEMICRO se escoge **PEDebug** y el microcontrolador correspondiente. Este modo es por defecto sólo para lenguaje ensamblador, por lo que solo aparece la opción de **Asm**.

New Project	×
Select project stationery:	
Project Stationery	
	•
. MON08 Target Interface	_
⊡~ PEDebug	
. Ē. AB	
. ∰ AS_AZ	
🖶 GP_GT .	
🖶 GP32	
Asm Asm	
⊕ GT16	
⊡ GT8	<b>↓</b>
OK Cancel	
OK Cancel	

Figura 2.

Cuando presione **OK** *Codewarrior* creará una carpeta con el nombre que le dio a su proyecto y creará un proyecto de ejemplo en lenguaje ensamblador.



Figura 3. Proyecto generado automáticamente por CodeWarrior.

Como el proyecto a realizar es en lenguaje C, se deben remover los archivos **GP32.asm** y **gpregs.inc** del proyecto, y adicionar la librería estándar de C y los archivos fuente necesarios. Para eliminar archivos del proyecto simplemente se seleccionan y se presiona la tecla Suprimir (esto no elimina los archivos del disco duro). Para adicionar archivos al proyecto se resalta la carpeta **sources** y en el menú se escoge: **Proyect** | **Add Files...** 

# Adicionar librería y archivo de arranque para trabajar en lenguaje C

Siempre se deben agregar dos archivos obligatoriamente: La librería estándar de C y el archivo Start08.c. Estos archivos se encuentran en las siguientes rutas:

```
\CodeWarrior HC08_V2.0\lib\hc08c\lib\ansi.lib.
```

\CodeWarrior HC08\_V2.0\lib\hc08c\src\Start08.c

En la figura 4 se observa como debe aparecer el proyecto después de agregar estos archivos



Figura 4. Archivos que se tienen que adicionar al proyecto.

#### Modificación del vector de interrupción de reset

Posteriormente se debe editar el archivo **default.prm**. Este archivo es muy importante porque define las zonas de memoria y los vectores de interrupción. En la figura 5 se indica donde esta ubicado en el proyecto. En este archivo se debe hacer lo siguiente:

Eliminar la línea	:	INIT Entry
Cambiar la línea	:	<b>VECTOR ADDRESS 0xFFFE main</b>
Por la línea	:	VECTOR ADDRESS 0xFFFE Startup

**\_Startup** es la rutina de arranque del microcontrolador, la cual está definida en **Start08.c**. La instrucción **VECTOR ADDRESS** permite definir los vectores de interrupción. Por lo tanto agregue todos los vectores de interrupción que use en su sistema. Más adelante se ilustrara con detalle la implementación de las rutinas de interrupción. En la figura 6 se muestra como debe quedar el archivo **default.prm**. Si lo desea puede descargar el archivo gp32.prm (<u>http://microe.udea.edu.co/cursos/ieo-944/files/hc08/gp32.prm</u>) que contiene información de todos los vectores de interrupción facilitando la programación de interrupciones.



Figura 5. Localización del archivo default.prm



Figura 6. Archivo default.prm después de ser editado

### Agregar archivos fuente al proyecto

El siguiente paso es agregar todos los archivos fuente correspondientes al proyecto. Para esto puede utilizar el menú **File** | **New...** como se muestra en la figura 7, o adicionar archivos ya existentes con **Proyect** | **Add Files...** 

Como en todo programa C, debe existir una función main(). En la figura 8 se muestra una plantilla mínima para el programa.

New	×
New Project File Object Text File	File name: main.c Location: jos\abaena\example08\sources
	Add to Project Project Targets:
<u></u>	Aceptar Cancelar

Figura 7. Crear nuevos archivos de código.



Figura 8. Función main()

Se recomienda ubicar todos los archivos fuente de un proyecto en la carpeta **sources**, la cual esta dentro de la carpeta del proyecto creado. Esto facilita el transporte del proyecto entre distintos computadores y evita cambios de código accidentales.

El proyecto de ejemplo es un sistema el cual prende y apaga un Led, donde los intervalos de encendido y apagado son programables por el puerto serial utilizando una aplicación tipo terminal. En la figura 9 se muestran todos los archivos fuente del proyecto en la carpeta **sources** y en la figura 10 el aspecto general del proyecto luego de adicionar los archivos. Estos archivos están disponibles en la página **http://microe.udea.edu.co/cursos/ieo-944/files/demoHC08** 

🔍 Explora	ndo - sou	irces							- 🗆 🗵
Archivo	<u>E</u> dición	⊻er	<u>İ</u> ra	<u>F</u> avoritos	<u>H</u> erramientas	Ay <u>u</u> da			- 10 C
	-	⇒	-	È	X	Þ		K)	$\sim$
Atrás	Ad	delante		Subir	Cortar	Copiar	Pegar	Deshacer	Eliminar
Dirección	🗋 D:\Us	uarios\	abaen	a\example08	\sources				•
Todas las (	carpetas					× Nombr	e	Tamaño	Tipo
		examp	ole08			🔺 💹 Dec	larations.h	1 KB	C header file
		ы 🔁	n			🗂 🌆 logp	o32.h	8 KB	C header file
		- 🛅 cr	nd			🛛 🚺 🚮 mair	n.c	2 KB	C source file
	. E	👝 e>	ample	08_Data		🔄 🌌 Star	t08.c	9 KB	C source file
		- 🗋 pr	m			🗂 🔝 term	nio08.c	2 KB	C source file
		🔁 so	ources						
		Fuzzy	comp						) E
5 objetos					18,9 KB (Espa	acio libre en 🚽	🗐 Mi PC		

Figura 9. Archivos fuente del proyecto.

example08.mcp		_	
NSM Generic	🄃 😽	🧭 🍃	× 🖻
Files Link Order Targets		Make	
💉 File	Code	Data 💐	<b>E</b>
👔 readme.txt	n/a	n/a	<b>x</b>
⊡ 🔄 Sources	72K	1K -	• 🔳
- 🚺 ansi.lib	73379	2015	
📲 main.c	242	5	• 🔳
Start08.c	166	9 -	• 🔳
🛄 termio08.c	37	0 -	• 🔳
🕀 🧰 Debugger Cmd Files	0	0	
🖃 🖂 Prm	0	0	
🛛 🚺 default.prm	n/a	n/a	
🛄 🛅 burner.bbl	n/a	n/a	
🕀 🧰 Debugger Project File	0	0	
			-
13 files	72K	1K	11

Figura 10. Archivos del proyecto de ejemplo

### Un proyecto de ejemplo

En esta aplicación se utiliza una interrupción de Output Compare para medir el tiempo de encendido y apagado del LED. Para implementar la rutina de interrupción se deben realizar dos pasos.

1. Se define la rutina de interrupción como una función que no recibe parámetros y que no retorna resultados *(void)*. Justo antes de la definición de la función se debe agregar la línea **#pragma TRAP\_PROC**. Esto le indica al compilador que esta función es una rutina de interrupción. En la figura 11 se observa este procedimiento

```
#pragma TRAP_PROC
void OutputCompare_Isr(void){
    //Aqui va el codigo de la rutina de interrupción
}
Figura 11. Definición rutina de interrupción
```

2. El siguiente paso es agregar esta función en los vectores de interrupción, para esto en el archivo **default.prm** se agrega, por cada rutina de interrupción, la línea:

# VECTOR ADDRESS [Dirección] [Nombre\_función]

Para el ejemplo anterior el archivo default.prm quedaría como se observa en la figura 12.

🗈 default. prm 📃 🗖 🗙
🚸 👻 {} 👻 M. 👻 🛅 👻 🖆 👻 Path: 🖸:\Documents and Settings\abaena\Mis docum\default.prm 🔷
✓* This is a .Prm file for the GP32*/
NAMES END
SEGMENTS RAM = READ_WRITE 0x0040 TO 0x023F; ROM = READ_ONLY 0x8000 TO 0xFDFF; END
PLACEMENT DEFAULT_ROM INTO ROM; DEFAULT_RAM INTO RAM; SSTACK INTO RAM; END
STACKSIZE 0x60
VECTOR ADDRESS 0xFFFE _Startup
I I I I I I I I I I I I I I I I I I I
Line 20 Col 1

Figura 12. Definición vector de interrupciones

Para la comunicación serial con la terminal se utilizaron funciones para I/O que vienen con las librerías del *Codewarrior*. Para poder utilizar estas funciones se debe agregar al proyecto el archivo **termio08.c** el cual define los registros del modulo SCI y puede ser encontrado en la ruta:

\CodeWarrior HC08\_V2.0\Examples\HC08\HC08 SIMULATOR\HC08 C\_Calc\sources

Además, se debe asegurar que el enlazado de este archivo se de antes que el de la librería estándar **ansi.lib**. Para hacer esto, en la ventana **Link order** se arrastra con el *mouse* la librería hasta la última posición tal como se ve en la figura 13.

prueba2.mcp		
Notement ASM Generic	- 🗈 😽	🤏 💺 🗎
Files Link Order Targets		
💉 File	Code	Data 🕊
🚺 readme.txt	n/a	n/a 🔳 📥
🚺 default.prm	n/a	n/a 🔳
📓 burner.bbl	n/a	n/a 🔳
📓 Postload.cmd	n/a	n/a 🔳
📓 Preload.cmd	n/a	n/a 🔳
📓 Reset.cmd	n/a	n/a 🔳
📓 Startup.cmd	n/a	n/a 🔳
📓 Setopu.omd	n/a	n/a 🔳
👔 project.ini	n/a	n/a 🔳
📓 Start08.c	166	9 • 🔳
🚺 main.c	242	5 • 🔳
🚺 termio08.c	37	0 • 🔳
💼 ansi.lib	73379	2015 🔳 🚽
13 files	72K	1K

Figura 13. Orden del enlazado del proyecto

# Compilación del proyecto

Después de tener todos los archivos del proyecto y actualizado los vectores de interrupción en el archivo default.prm se procede a compilar y enlazar el proyecto. Para esto seleccione **Proyect** | **Make.** 

Después de compilar los archivos CodeWarrior le dará un informe de errores, advertencias y mensajes. Si no obtiene ningún error, se puede pasar a la simulación.

**NOTA:** A pesar de que no haya errores de programación el enlazador muestra el error observado en la figura 14. Esto se debe a que el proyecto era inicialmente para ensamblador y no para C. Para corregir esto, se selecciona **Proyect** | **Debug**. Inmediatamente aparecerá el mensaje de la figura 15. Al presionar **Yes**, se recompilan todos los archivos y se abre el simulador. A partir de este momento, si se vuelve a reconstruir el proyecto con **Proyect** | **Make**, no volverá a aparecer el error mencionado.



Figura 14.



# Simulación del proyecto

En la figura 16, se muestra la ventana de simulación de CodeWarrior. En ella se pueden observar el contenido de los registros, memoria, variables globales y locales y el código fuente en C y en ensamblador. En la ventana **Command** se pueden escribir las instrucciones de simulador de PEMICRO. Estas instrucciones también se pueden ejecutar desde el menú **PEDebug**.

Finalmente, solo queda grabar el programa en el microcontrolador. El archivo generado se llama **AsmGeneric.S19**, el cual esta ubicado en la carpeta **bin** del proyecto. En la figura 17 se observa el programa ya funcionando.

🐍 True-Time Simulator & Real-Time Debugger D:\Usuarios\abaena\exa	mple08\project.ini 📃	I×
<u>File View Run P</u> EDebug <u>C</u> omponent <u>M</u> emory <u>W</u> indow <u>H</u> elp		
S Source		×
D:\Usuarios\abaena\example08\sources\Start08.c	: 276Startup	
*/	SET	
#ifdefELF_OBJECT_FILE_FORMAT	LDA 0x80A6	
DisableInterrupts; /* in HIWARE format, this is done i	in the BIT #0x02	
#endif	BNE *+6 ;abs = 8098	
Ior (;;) { /* forever: initialize the program; call t	TXS	
		-
Procedure	E Register	×
	HC08 CPU Cycles: 4936493 Auto	
Startum ()	A 2E	
	HX 94 SP FF	
A Data	ITIXI SR 68 Status VHINZC	
	PC 808C	
Auto Symb		
	,	
	Memory	<u>×</u>
	Auto	
	0000 uu uu uu uu 00 00 00 00 uuuu	
Lommand	0008 uu 00 00 00 00 u	—
STEPPED	$\square 0010 28 08 uu 00 00 00 00 00 (.u)$	
STEPPED		
STEPPED	0028 00 uu uu 20 00 00 FF FF .uu	
STEPPED	0030 00 uu uu 00 uu uu 20 00 .uu.uu .	
Reset command file is disabled.	0038 00 40 40 01 1F uu 0000u	
RESET		
		-
For Help, press F1	HC908GP32 A user initiated RESET has occured.	

Figura 16. Sistema de simulación.

🗞 hc08 - HyperTerminal	- 🗆 🗵
<u>Archivo Edición V</u> er <u>L</u> lamar <u>T</u> ransferir Ay <u>u</u> da	
Ejemplo para HCO8	<b>^</b>
Ingrese tiempo ON del led (ms)(3448Max): 500	
Ingrese tiempo OFF del led (ms): 500	
Ingrese tiempo OFF del led (ms): 1000	
Ingrese tiempo OFF del led (ms):	
00:01:24 conectado ANSI 19200 8-N-1 DESPLAZAR	MAY /

Figura 17. Programa final funcionando en el microcontrolador.