CURSO DE PROGRAMACIÓN DE MICROCONTROLADORES EN LENGUAJE C/C++

M. Barrón, J. Martínez
Dpto. de Ingeniería de Sistemas y Automática
Escuela Universitaria de Ingeniería Técnica Industrial
Universidad del País Vasco
Avenida de Otaola, 29
20600 Eibar (Guipúzcoa)
Tfno: (943)108444 - Fax: (943)103196
E-mail: ispbarum@sb.ehu.es

RESUMEN. En este artículo se describe brevemente el hardware, software y material didáctico desarrollado, así como el método de trabajo utilizado en la creación y depuración de software en lenguaje C/C++, para aplicaciones que incorporen un microcontrolador compatible en software con la familia de procesadores Intel 80x86.
La generación de código para estos microcontroladores se realiza con los mismos entornos disponibles en los PCs para la elaboración de aplicaciones DOS o WINDOWS. Estos entornos, creados por Borland o Microsoft, dinamizan el proceso de producción de software, al integrar compiladores altamente optimizados con eficaces depuradores y con otras herramientas de desarrollo.

1. INTRODUCCIÓN

La experiencia de la programación en lenguaje ensamblador con diversos microprocesadores y microcontroladores, nos ha ido haciendo patente la necesidad de programar en lenguajes de alto nivel. De esta forma se generan funciones que pueden reutilizarse en proyectos sucesivos, se escriben programas mejor documentados y se consigue una mayor productividad en la elaboración del software. Además, si por cualquier razón nos vemos obligados a utilizar otro microcontrolador, el cambio resultará sencillo siempre que se disponga de un compilador para el nuevo microcontrolador.
Por contra, la creencia de que la programación en lenguajes de alto nivel produce un aumento del código y una pérdida en velocidad de ejecución de los programas no resulta muy acertada cuando se dispone de compiladores optimizados.
La Escuela Universitaria de Ingeniería Técnica Industrial de Eibar posee, desde hace años un sistema para el estudio del microcontrolador Intel-8051, que permite la programación en lenguaje ensamblador y en lenguaje C. Teniendo cubiertas las necesidades en el campo de los microcontroladores de 8 bits, se pensó abordar el campo de los microcontroladores de 16 bits, buscando un método de trabajo que permitiera alcanzar los objetivos expuestos en el siguiente apartado.

2. OBJETIVOS

El sistema utilizado en el curso de programación de microcontroladores en lenguaje C/C++, además de ser eficiente y económico, deberá:

- Capacitar al estudiante en el desarrollo de software para sistemas basados en microcontroladores utilizando lenguajes ensamblador, C y C++ .
- Permitir la depuración de código en lenguajes de alto nivel y en ensamblador.
- No exigir el uso de emulador.
- Ofrecer la posibilidad de añadir circuitería adicional para las clases y para los Proyectos Fin de Carrera.

3. DESCRIPCIÓN DEL SISTEMA

Debido a que el curso tiene aspectos teóricos y prácticos, se desarrolló material para el laboratorio (hardware y software), y material didáctico.

3.1 Hardware

Se trata de una única tarjeta provista de:

- Microcontrolador NEC-V25 operando a 16Mhz.
- Un chip de memoria EPROM de 8, 16, 32, 64 o 128 Kbytes.
- Un chip de memoria EEPROM a través de bus serie I²C.
- Un chip de memoria SRAM de 128 o 512 Kbytes.
- Reloj en tiempo real.
- Batería de litio para la SRAM y el chip de Reloj.
- Dos puertos serie RS-232 o RS-485.
- Puerto paralelo compatible Centronics.
- Puerto de 8 entradas opto-aisladas.
- Puerto de 8 salidas opto-aisladas.
- Amplificador de audio de 1/2 watio.
- Conexión directa para displays LCD de hasta 4*40 caracteres.
- Conector para teclado matricial de hasta 16 teclas.
- Conector para teclado de PC.
- Expansión serie a través de bus I²C.
- Bus de expansión para tarjetas compatibles PC (Bus XT).

De los posibles microcontroladores compatibles con la familia de procesadores Intel 80x86 (80C166/188, 80386EX de Intel; Am186EM/188EM de AMD y los miembros de la serie V de NEC), se optó por el NEC-V25 debido a que:

- Integra en un solo chip un buen número de periféricos [1]: circuito generador del reloj del sistema, 16Kbytes de ROM, 256 bytes de RAM interna, controlador DMA de dos canales, dos puertos serie con sus generadores de velocidad de comunicación, un contador base de tiempos, dos temporizadores, un controlador de interrupciones, tres puertos de entrada/salida de 8 bits, y un puerto de 8 entradas a través de comparador, con tensión de comparación programable en 16 niveles.
- Es compatible en software con la familia de procesadores Intel 80x86, lo cual significa que para la programación del NEC-V25, pueden utilizarse los mismos compiladores que se emplean con los ordenadores personales.

- Dispone de ocho bancos de registros, lo que facilita la realización de multitarea y el procesado rápido de interrupciones. Además, las interrupciones que requieren un procesado simple, tal como el movimiento de un dato entre memoria y un registro de función, pueden atenderse mediante macroservicios, que no exigen la ejecución de ningún software.

El hecho de que el microcontrolador elegido posea un bus de datos externo de 8 bits, hace que los equipos desarrollados con él, resulten menos costosos que los equipos construidos con un microcontrolador provisto de un bus de datos de 16 bits, debido a que solo se necesita utilizar la mitad de chips de memoria, y además se ahorra un buffer para el bus de datos. Lógicamente la velocidad de ejecución de código se resiente. Sin embargo, las prestaciones no quedan reducidas a la mitad, sino que se aproximan al 64% [2].

3.2 Software

Las funciones de la librería del compilador utilizado (Borland C++ 3.1) que dependen del hardware del PC, no sirven para la tarjeta diseñada y deben ser reescritas. Seguidamente se ofrece un listado de los prototipos de algunas de las funciones de mayor nivel que se han elaborado para el desarrollo de aplicaciones con la tarjeta de prácticas:

- Funciones de manejo del bus I²C

```
void iic_init(void);
int  putchar_IIC(UCHAR dir_esclavo, UCHAR dato);
int  getchar_IIC(UCHAR dir_esclavo, UCHAR far *destino);
int  puts_IIC(UCHAR dir_esclavo, int num_bytes, UCHAR far *dir_org);
int  gets_IIC(UCHAR dir_esclavo, int num_bytes, UCHAR far *dir_dest);
```

- Funciones de teclado

```
void init_kb_at(void);
void init_kb_16(BOOL click);
int  kbhit(void);
int  getch(void);
```

- Funciones para el manejo de los puertos serie (UART)

```
void openUART(int Numport,long baud,int paridad,int stopbits,
              int numbits,int *codigo_error );
void closeUART(int Numport);
void checkUART(int Numport, UCHAR *c, int *codigo_error);
void getsUART(int Numport, UCHAR *str, int *codigo_error);
void putcharUART(int Numport, UCHAR c, int *codigo_error);
void putsUART(int Numport, UCHAR *str, int *codigo_error);
```

- Funciones para el manejo del display LCD

```
void InitDisplay(int tipo_lcd, UCHAR dir_iic);
int  _putch(int ch);
void putchar_LCD(char direc, char dato);
void ScrollLcd(void);
void clreol(void);
void clrscr(void);
void delline(void);
void gotoxy(int x, int y);
```

- Funciones para el manejo de la impresora
  ```
  void printer_init(void);
  int  putchar_printer(UCHAR dato);
  int  puts_printer(UCHAR * pcadena);
  ```
- Funciones para la generación de música y sonido
  ```
  void sound(unsigned frecuencia);
  void nosound(void);
  void interpreta(UCHAR *pint);
  void acorde(unsigned char duracion, ...);
  void delay(int num_msgs);
  ```

3.3 Material didáctico

El material didáctico elaborado se compone de:

- Manual del microcontrolador NEC-V25.
- Manual descriptivo de la tarjeta de prácticas y de todos sus chips.
- Manual con ejemplos de programas que utilizan todas las características del microcontrolador y de sus periféricos.
- Transparencias.

4. MÉTODO DE TRABAJO

El método de trabajo es el típico bucle edición-compilación-depuración, que se repite hasta que el programa se comporta de la manera deseada. La depuración se realiza sobre la tarjeta de prácticas con todo el hardware que se le haya podido añadir, para una aplicación específica. Mientras dura el proceso de depuración, los programas se descargan en la memoria RAM de la tarjeta. Cuando el bucle de desarrollo concluye, el programa resultante se almacena en una memoria EPROM, que será la que tome el control al aplicar la tensión de alimentación.

Las herramientas de programación que utilizan los alumnos son:

- Editor de textos.
- Compilador C/C++ para PC (Borland C++ 3.1).
- Localizador de programas ".EXE" (Paradigm LOCATE)
- Depurador remoto (Paradigm DEBUG)

Cuando el MS-DOS carga en memoria RAM un programa con extensión ".EXE", modifica los valores de los segmentos de programa para adaptarlos a la dirección de memoria donde van a residir. Este proceso se denomina ajuste o localización [3]. El programa localizador [4] es el encargado de transformar un programa ".EXE", en un programa preparado para correr en direcciones fijas de la memoria EPROM de un microcontrolador.

Para depurar software en la tarjeta de prácticas, se necesita unir un puerto serie de ésta, con un puerto serie del PC. El depurador remoto [5] es un programa que corre en el PC y controla la tarjeta de prácticas, permitiendo entre otras cosas:

- Cargar el programa a depurar en la memoria RAM de la tarjeta de prácticas.
- Ejecutar el programa a toda velocidad, con o sin puntos de parada.
- Ejecutar el programa paso a paso, en C o en ensamblador.
- Visualizar el código fuente, la memoria, la pila, los registros, los periféricos, los valores de las variables y los puntos de parada del programa.

- Cambiar el valor de las variables.
- Seleccionar las variables a examinar, mientras el programa esté en ejecución.

Para hacer posible el diálogo entre el PC y la tarjeta de prácticas, es preciso que ésta disponga de un *Kernel* o pequeño programa almacenado en memoria EPROM. Este programa ocupa menos de 8Kbytes y se genera mediante la utilidad TDREM [6] de Paradigm Systems.

5. DESARROLLO DEL CURSO

La limitación horaria del plan de estudios de Ingeniería Técnica, en la Especialidad de Electrónica, no permite el estudio de dos microcontroladores diferentes. Todos los alumnos de esta Especialidad estudian el microcontrolador Intel-8051, utilizando un sistema similar al descrito en este artículo para el microcontrolador NEC-V25. La experiencia docente con el sistema descrito se limita a, tres cursos intensivos de unas 40 horas de duración, impartidos a estudiantes voluntarios fuera del horario lectivo, y al seguimiento de los Proyectos Fin de Carrera y de las aplicaciones realizadas con este sistema.
A cada curso han asistido una media de 15 alumnos, a razón de dos personas por cada puesto de trabajo. Las clases se impartieron en el laboratorio, utilizando transparencias y proyectando las imágenes de la pantalla del ordenador. En los cursos no ha sido necesario explicar el lenguaje C, debido a que los alumnos de la Escuela lo estudian en cursos anteriores, no obstante se considera conveniente incluir en las primeras jornadas, un rápido repaso al lenguaje, insistiendo en los aspectos relacionados con el manejo de los punteros (*near* y *far*) y las estructuras.

6. CONCLUSIONES

En general, todos los asistentes a los cursos manifestaron una buena aceptación de los mismos, debido a su orientación práctica y a las herramientas software utilizadas, que por venir del mundo del PC, superan en calidad y eficacia a las herramientas utilizadas con otros microcontroladores no compatibles en software con la familia Intel 80x86.
Con relación a las aplicaciones realizadas con la placa de prácticas, se ha observado una drástica reducción en el tiempo de desarrollo, en comparación a las aplicaciones realizadas en lenguaje ensamblador, debido principalmente al uso de funciones de librería y de las funciones creadas para atender a los posibles periféricos de la tarjeta.

7. REFERENCIAS

[1] "V25/V35 Family V-Series 16-bit Microcomputers User's Manual". *NEC*

[2] Naro, R. "Characterizing Processor Performance". *Circuit Cellar Ink*, #57, 1995

[3] Allison, C.B. "ROMLDR, an Embedded System Program Locator". *The C Users Journal*, Vol. 12, nº 3, 1994

[4] "Paradigm LOCATE. Embedded System Software for Intel 80C186 and NEC V-Series Microprocessors". *Paradigm Systems*.

[5] "Paradigm DEBUG/RT-V25 User's Guide". *Paradigm Systems*.

[6] "Paradigm DEBUG/RT and Turbo Debugger Remote Debugging Kernel". *Paradigm Systems*.