SYSTECH J.Schnyder GmbH

Schlifweg 30
CH-4106 Therwil
Telefon ++41 (0)91 827 15 87
www.systech-gmbh.ch

EBS08

A simple Operating System for HC(S)08 Microcontroller

V 0.7

NEW TIMING!!

Overview for the version 1.3

Content

Preface

The applications for microcontrollers in todays units are increasing constantly. Many times the tasks are very similar, so that it make sense to use a simple operating system with some standard functions.

The operating system described below allows to implement communication tasks, periodic tasks an more with the aid of a simple task manager. It is written in assembler to guarantee a high performance and a small ROM/RAM usage.

Development Environment

The version described below was written for the Metrowerks Codewarrior, but runs with small changes with other Development environments.
The concept is based on include files (*.mmw). This means, that the user has to include the used modules and variables in the main file. This has the vantage, that there is no need for a version management and the whole structure is kept simple, witch is important for beginners.

Basic System

The basic system consist of main file, TIC routine, Task manager und vector definitions.

- the main file contains all the used modules (included) and the declaration of the variables and the INI section sets up the basic functions and variables for the OS and also for the user functions.
- the TIC routine is microcontroller dependent and generates the periodical timer interrupts.
- in task manager the user has to insert his functions for the desired actions the task manager also generates the TICs used to manage the timing. The task manager is available in three versions: fast, slow and compact. For more information see the file TICs.pdf.
- the vector block contains the interrupt vectors

The user writes his desired functions and tasks in his own modules and controls the program flow with the aid of the task manager.

Additional Modules

There are additional modules:
- mathematics module
- conversion modules
    - BIN-BCD
    - BCD-ASCII
    - BCD-7 segment
    - SINGLE (floating-point) to LONG
- keyboard module for 4x4 matrix keyboards
- LCD module (4-bit mode)
- clock module with second, hour, month, year and weekday
- segment multiplexing module for displays with 8 digits and 8 segments
- EEPROM programing routines
- communication routines
    - serial communication
    - CRC routines

- stepper motor routines for full/half step 1- or 2-phase mode
- and more (ask: -> info@systech-gmbh.ch)

Description of the System

Main File

the main file is used to declar constants and variables of the program, and to include the used. The included INI section is used to initialize the system. The different subsystems must be initialized correctly including the TIC module, the watchdog and the flags for the task manager. The user has also to take care about his own variables and used subsystems.

TIC Module

The TIC module generates the flag TIC_F1 in the CORE_F0 byte used by the task manager.
The TIC interval is based on the TIC constant defines in the main file.
The TIC interval depends on the microcontroller clock and on the timer hardware used.

i.e. is the clock rate of the Qx-family 12.8 MHz (nominal). But the system clock is four times slower (3.2MHz). That means that the cycle time is 321 ns. For a TIC interval of 1ms wee need 3200 cycles.

So we need to set the TIC constant

TIC EQU 3200

The constants TICH and TICL are calculated automatic.

The system contols the following counters and flags:

Z_CORE	the core timer witch is incremented every TIC. It can be used to calculate times between events and bit0 can be used as F_TIC2.
CORE_F0	The byte that contains the Flags. The TIC routine has only to set F_TIC the rest is done by the task manager.
F_TIC1	is the most important flag for the task manager since the task manager starts a new scan if the flag changes from “0" to “1". This flag will be cleared by the task manager. (All other flags are generated by the task manager itself)

The TIC module has to be adopted for the MPU used, because not all MPU’s are using the same timer module. Sometimes there are more than one TIC module present, so the user can choose the more suitable one for his application.

TASK Manager

The task manager controls the program tasks of the system. With the aid of the different flags the user can control and mange the program flow. Normally the tasks are signaling there status with

the task manager flags (TASK_F1).

I.E:
- wait for a event
- executing a task
- action required

First the task manager waits until F_TIC1 in the CORE_F0 byte is set. Then it generates (calculates) the flags listed below:

The following Flags are signaling the start of a new time slice and are active for just one TIC.

	active every
F_TIC4	4. TIC
F_TIC8	8. TIC
F_TIC16	16. TIC
F_TIC32	32. TIC
F_TIC64	64. TIC
F_TIC128	128. TIC
F_TIC256	256. TIC

Then he resetts the watchdog and then the different tasks where executed. At the end of the task list the manager jumps to his loop waiting for the next TIC_F1 event.

The single tasks preform the actions needed. With the aid of the TIC flags the task can decide if or if not to execute the action in a certain time slice. It is possible to execute all tasks sequentially or to execute a task with a high priority alone simply by jumping at the end of the task manager after executing the action.

Attention:
The user has to watch, that the maximum execution time is not longer than the TIC period.
In case of longer execution times the system runs anyhow; but it is possible that the task manager does not execute all desired tasks!

If there is a request for more flags, they can easily added by defining them in the main file. Variables containing Flags are normally ending with ...._F or ...._FLG and the flags themselves begin with F_....

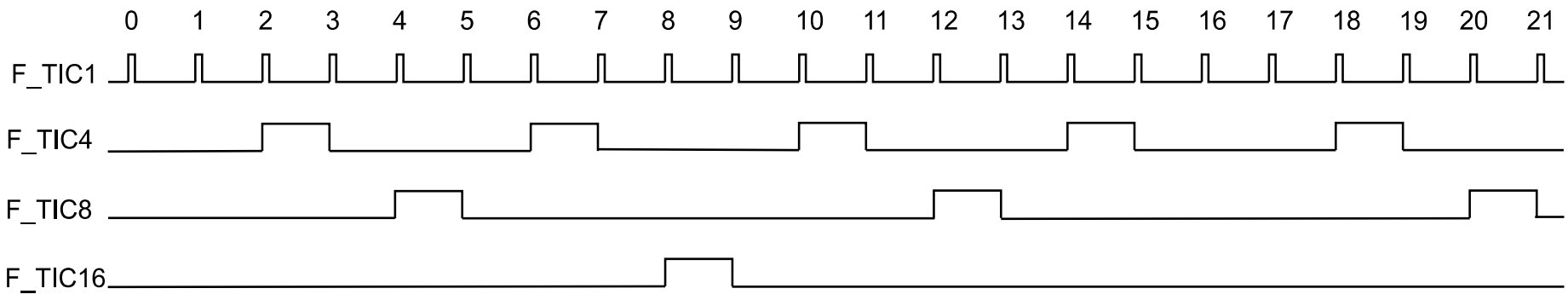
The tasks are named in the task manager begining with T_.... and ending with T_....E.

Vector-Block

The vector block contains the used interrupt vectors.
For a simple system there has to be only the timer vector and the reset vector.
If more interrupt vectors are needed, they have to be added with their names in the vector block.
In the EBS08 interrupt routines are named with I_ .... .
Vectors not used were usual initialized to point to a DUMMY interrupt routine containing only RTI.
This makes sure, that the program is not hanging.

Attention:
The reset vector is defined in the PRM file for the Code Warrier! Therefore this vector has to be uncommented in te vector table!

EBS08-Timing



Naming-Convention

general:		example
names of variables are written in capitals		VAR1
Subroutines begin with the module name		SUB1_AKT1
and are ending with a label with the same name+"E"		SUB1_AKT1E

whereas SUB1 is the name of the module and AKT1 the name of the routine.

Exceptions are the interrupt routines and the sections in the task manager.

Variables and constants:

countes	Z_*name*	Z_CORE
flag variables	*name*_F	CORE_F
flags	F_*name*	F_TIC1
interrupt routines	I_*name*	I_TIC
task manager sections	T_*name*	T_AKT1

Changing the EBS08 to a other family member

Creating a new project

Open CodeWarrier IDE
Select **File** then

New

HC(S)08 New Project Wizard

set the locaion of the project to: **c:\hc08\projects**
and name the project **EBS08_*microcontroller_name***

klick **OK**

select the controller ***microcontroller*** then **Next** (Avanti or whatever)

unselect C and select **Assembler** then **Next**

select **Absolute Assembly** then **Next**

select **P&E Full Chip Simulation, P&E Hardware Debugging** and **SofTec Microsystems** then **Finish** (Fine ....)

The new project will be created now.

Prepairing the new project

Open the desktop (or Total Commander) and copy the files found in the folder with the original EBS08 i.e:
c:\hc08\projects\EBS08_GB32\sources
to
c:\hc08\projects\EBS08_*microcontroller_name*\sources
allowing the overwriting of the file main.asm.

Then modify the main.asm file for the new family member (normaly you have only to change the memory definition file *.mem)

Links:

For more information see also the tutorials and examples on:

www.systech-gmbh.ch