SYSTECH J.Schnyder GmbH

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

EBS08C

A simple Operating System for HC(S)08 Microcontroller

V 0.2

The “C” Version!!

Overview for the version 1.1

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 the templates of CW 5.1. This means, that the user has to include the header files used in the main file and to add the modules in the project.

Basic System

The basic system consist of main file and the tic files (tic.c and tic.h).

- the main file is ready to run (a simple task, usually a increment of a output port).
- the tic module is prepared for three different TIC algorithms and for different timer interrupt arrangements. Depending on the MPU used, TIC interrupts can be generated i.e. by RTI, By a timer in MOD mode or by timer in output compare mode!
- the user needs only to comment/uncomment the flags in the tic.h file (more information below)

Additional Modules

There are additional modules: (at the moment as .asm files only)
- 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
    - MIP Bus Protocol (RS485) MAXXON-Motors
- 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 prepared and ready to use! Users may add the code needed in form of functions or can put the code directly in the task manager section

Task Manager Section

The task manager controls the 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 their 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 F0_TIC1 in the CORE_F0 byte is set. After feding the watchdog, the TIC_FLAGS routine 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
Fx_TIC4	4. TIC
Fx_TIC8	8. TIC
Fx_TIC16	16. TIC
Fx_TIC32	32. TIC
Fx_TIC64	64. TIC
Fx_TIC128	128. TIC
Fx_TIC256	256. TIC

x = 0 or 1 depending on the actual phase if Z_CORE (the core counter) contains a even value phase0 is active, on odd values phase1.

Now the different tasks where executed.
At the end of the task list the program begins at the start.

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.

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_....

TIC Module (tic.c and tic.h)

The TIC module generates the flags needed in the task manager section of the main file.

The file is prepared for the generation of the task following flags:

PHASE0	PHASE1
F0_TIC4	F1_TIC4
F0_TIC8	F1_TIC8
F0_TIC16	F1_TIC16
F0_TIC32	F1_TIC32
F0_TIC64	F1_TIC64
F0_TIC128	F1_TIC128
F0_TIC256	F1_TIC256

There are implemented thee algorithm for the flag generation.
The compact version is optimized for code size,
the slow(er) version uses a 16 byte table to calculate the flags and
the fast version uses a 128 byte table to do this. The fast version has the benefit to do the calculation fast (obviously) and vith a minimum of variations in the timing.

For more information on TIC's you may have a look at How to generate TIC's on:

http://www.systech-gmbh.ch -> EBS08 -> DOC's

to select the flag mode simply uncomment one of the corresponding line in the tic.h file:

```
#define tic_g_compact          // generates the TICs with a compact code
// #define tic_g_slow          // generates the TICs with a small table
// #define tic_g_fast          // generates the TICs fast with a bigger table
```

Attention! Make shure to comment out the lines not used!!! Only one tic_g_???? lable may be defined!

The constant TICS (in the tic.c file) is used to adjust the time between two TIC's. Normally the time is 1ms. The user may change it, but has to take care not using too short times. For certain clock sources the TICS constant is not used (i.e. for the RTI mode).

The different clock sources can be selected by uncomenting the needed version int the tic.h file

```
#define tic_m_mod      // tic with TIM counter in modulo mode
// #define tic_m_oc    // tic with TIM counter in output compare mode
// #define tic_m_rti   // tic with RTI interrupt
```

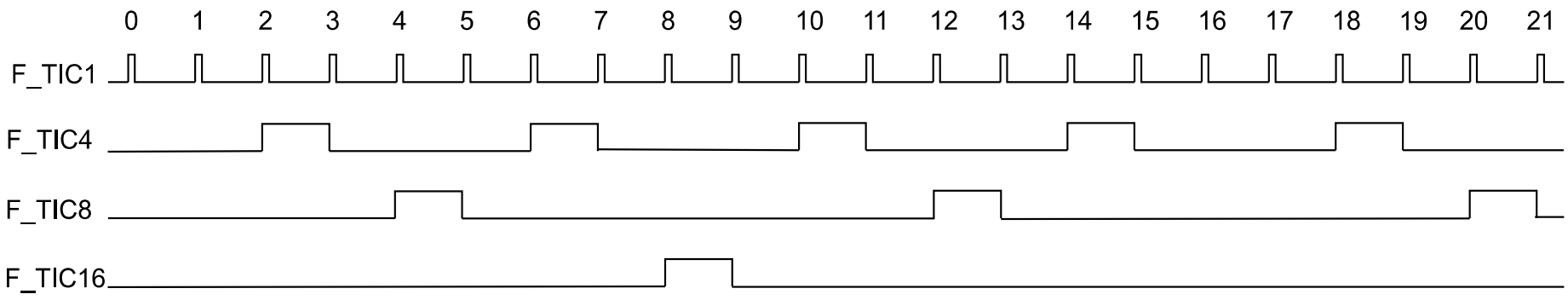
Attention! Also you must make shure to have enabled only one TIC clock mode!!

Note: For some MPU's there may be different clock sources available!

The I_TIC interrupt routine is very short; It prepares the clock source for the next time out period,

increments the core counter Z_CORE and sets bit 0 in the CORE_F0 variable!

EBS08-Timing



Naming-Convention

general:		example
names of variables are written in capitals		VAR1
Subroutines begin with the module name		SUB1_AKT1

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

Changing the EBS08 to a other family member

Creating a new project

Simply copy the directory of the template you need (i.e. *EBS08C_QG8_51_11*)
to a new directory with the name you need.
In the new directory rename the *???????.mcp* file with the name you have chosen for the project.
To finish the preparation delete the *???????_Data* subdirectory.

If you need a specific TIC clock source or an other the flag generation mode you may change this in the tic.h file. The duration of the TIC period can be changed in the tic.c file (TICS).

Now you can insert your code in the task manager section, add modules.

Good Luck!

Links:

For more information see also the tutorials and examples on:

www.systech-gmbh.ch