



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 HC08 Microcontroller

V 0.5

Overview

Content

Preface	2
Development Environment	2
Basic System	2
Additional Modules	2
Description of the System	3
Main File	3
TIC Module	3
INI Routine	4
TASK Manager	4
Vector-Block	5
EBS08-Timing	6
Naming-Convention	7
For more information see also the tutorials and examples on:	7

Preface

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

The operating system described below allows to implement communication tasks, periodic tasks and 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 advantage, that there is no need for a version management and the whole structure is kept simple, which is important for beginners.

Basic System

The basic system consists of main file, TIC routine, INI block, Task manager and vector definitions.

- the main file contains all the used modules and the declaration of the variables
- the TIC routine is microcontroller dependent and generates the periodical timer interrupts.
- the INI block is used to set up the basic functions and variables for the OS and also for the user functions.
- in task manager the user has to insert his functions for the desired actions.
- 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 programming 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

Description of the System

Main File

the main file is used to declare constants and variables of the program, and to include the used

TIC Module

The TIC module generates the flags used in 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 we need 3200 cycles.

So we need to set the TIC constant

```
TIC EQU 3200
```

The constants TICH and TICL are calculated automatic.

The system controls 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_F in this byte you find all the F_TICx flags.

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 has to be cleared form the task manager. (All other flags should be left untouched)

The following Flags are signaling the start of a new time slice and are activ 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

The TIC module has to be adopted for the microcontroller used, because not all are using the same timer module.

INI Routine

The INI file is used to initializing 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.

TASK Manager

The task manager controls the program tasks of the system. With the aid of the different flags the user can control and manage the program flow. Normally the tasks are signaling their status with the task manager flags (TASK_F1).

I.E:

- wait for an event
- executing a task
- action required

How it works:

The task manager waits in a loop until the TIC_F1 flag in C_FLAGS changes from "0" to "1". After resetting the watchdog the different tasks are executed. At the end of the task list the manager jumps to his loop waiting for the next TIC_F1 event.

The single tasks perform the actions needed. With the aid of the TIC flags the task can decide if or if not to execute the action. 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; it is possible that the task manager does not execute all desired tasks!

If there is a request for more flags, they can easily be added by defining them in the main file. Variables containing flags are normally ending with_F and the flags themselves begin with F_. In the example two flags are defined named F_AKT1 and F_AKT2 in the variable TASK_F.

The tasks are named in the task manager beginning 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.

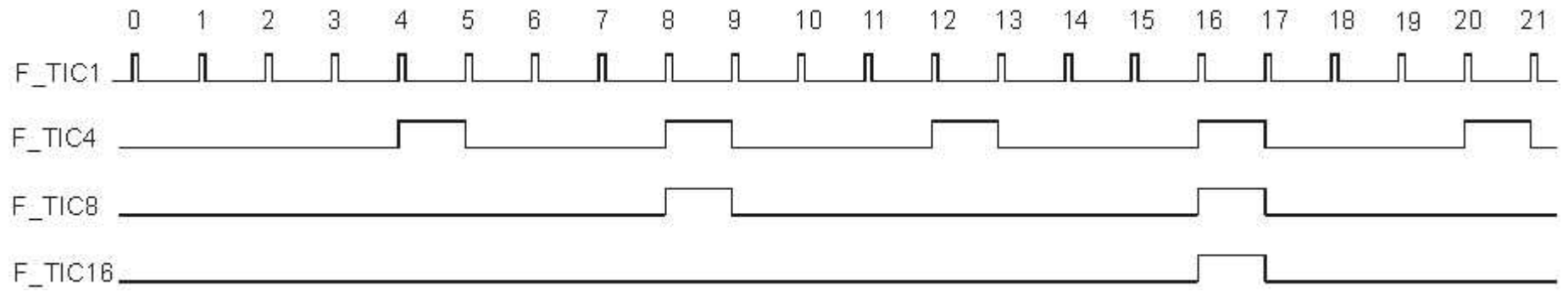
If more interrupt vectors are needed, they have to be added with their names in the vector block.

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.

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

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

Variables and constants:

countes	<i>Z_name</i>	Z_CORE
flag variables	<i>name_F</i>	CORE_F
flags	<i>F_name</i>	F_TIC1

interrupt routines	<i>I_name</i>	I_TIC
task manager sections	<i>T_name</i>	T_AKT1

For more information see also the tutorials and examples on:

www.systech-gmbh.ch