TWR56F8400 BLDC Motor Control with Hall Sensor Lab

1, Run the BLDC Motor demo with a Graphical FreeMaster Interface

This lab shows how to run a BLDC motor with hall-effect sensor, which is a basic BLDC motor control application controlled by two buttons (SW1 and SW2) and/or FreeMaster graphical interface.

TWR56F8400 BLDC motor control with hall sensor lab illustrates:

- Simple MC56F84789 motor control application
- How to configure Cyclic ADC, eFlexPWM, XBar, QuadTimer, GPIO, HSCMP, 6-bit DAC, and SCI peripherals for a basic motor control application
- How to implement FreeMaster driver with PE components
- How to implement PI controller with PE components
- How to use XBAR to synchronize ADC sampling with PWM trigger output
- How to configure synchronization among eFlexPWM sub-modules with sub-module0 as master
- How to use PE component events for hall sensor edge detection, button status detection and speed control interrupt timer
- ADC channels offset calibration at start-up process
- Usage of HSCMP and 6-bit DAC for over-current fault protection
- Advanced features of FreeMaster – recorder, scope, watch window and graphical control interface to monitor motor variables and control motor status

If connected via FreeMaster, speed, voltage, motor current and work status can be observed.

As well the following parameters can also be set:

- Desired motor speed by clicking on speed gauge
- Motor current limit by clicking on motor current gauge. By default the limit is set to 0.5 amps
- Write a “1” to value field of FAULTCLR variable to clear all application fault flags when there is any application fault occurred

This demo requires APMOTOR56F8000e motor control board (not delivered together with TWR-MC56F8400 tower system module)

2, Demo Setup

2.1 TWR-MC56F8400 Board

J7 – place a shunt onto 1-2 position to supply 3.3V from APMOTOR56F8000e board

J4 – place a shunt onto 2-3 position so that IRQ1 button (SW1) is connected to GPIOF6

J5 – place a shunt onto 2-3 position so that IRQ0 button (SW2) is connected to GPIOF7

J8 – place a shunt onto 2-3 position so that RXD0 (GPIOF8) is connected to JM60 (OSJTAG) TXD1 pin

J9 – place a shunt onto 2-3 position so that TXD0 (GPIOC2) is connected to JM60 (OSJTAG) RXD1 pin

2.2 APMOTOR56F8000e Motor Control Board

JG1 – place all three jumpers onto position of HALL

Plug TWR-MC56F8400 40-pin MOTOR J501 Connector into P1 40-pin header on APMOTOR56F8000e board with aligning pin1 from the P1 connector on the APMOTOR56F8000e motor control board with pin1 of the MOTOR J501 connector on the TWR-MC56F8400.

Connect P2 on APMOTOR56F8000e to 9V DC input for system power.

Note: the input is used directly as the motor DC bus voltage, so please use 9V input from damaging motor.

The APMOTOR56F8000e board will power the TWR-MC56F8400 board when the two boards are plugged together.

Note: don't use the power jack on the TWR-MC56F8400 board for this demo.

2.3 Application User's Control Interface

Motor can be controlled from buttons on TWR-MC56F8400 board (SW1 and SW2), or/and FreeMaster (preferred).

Possible application faults: over-current, over-voltage, and under-voltage. In order to identify a source of application fault, please use FreeMaster tool, which communicates with the target TWR-MC56F8400 via serial communication interface (SCI0).

Note: make sure that FreeMaster is installed on your PC before opening the FreeMaster project, and JM60 OSJTAG driver is installed before setting up the connection between FreeMaster and TWR-MC56F8400 target board.

2.4 Download Code into Target

Launch CodeWarrior v10.2, open “TWR56F84789_BLDC_HALL\.project” through drag & drop or “Import…” command in “File” menu.

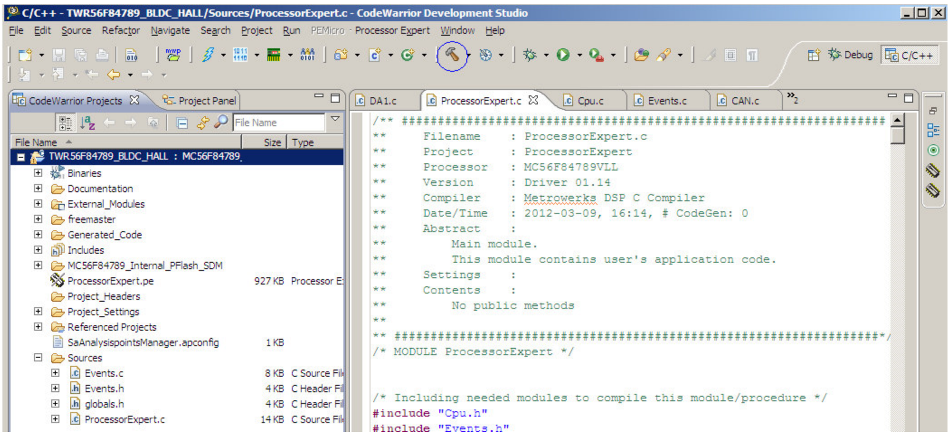
1, Plug APMOTOR56F8000e board with TWR-MC56F8400 board together

2, Power APMOTOR56F8000e board

3, Plug USB cable to mini-B USB connector

4, Plug USB cable into one of free USB ports in your PC

5, Click project tree of CodeWarrior project window to enable selection, then click the hammer icon of “build” command (blue circle in Fig.1) to build demo project

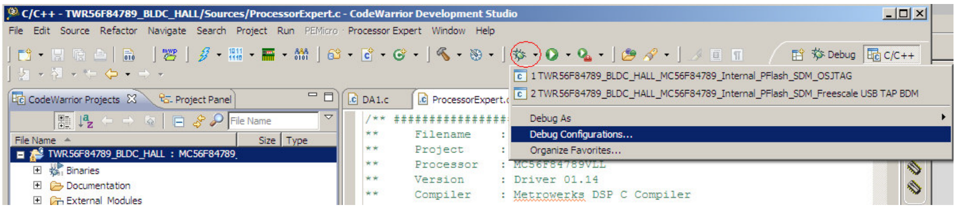

C/C++ - TWR56F84789_BLDC_HALL/Sources/ProcessorExpert.c - CodeWarrior Development Studio
File Edit Source Refactor Navigate Search Project Run PEMicro Processor Expert Window Help
Debug
C/C++
CodeWarrior Projects
Project Panel
File Name
Size
Type
TWR56F84789_BLDC_HALL : MC56F84789
Binaries
Documentation
External_Modules
freemaster
Generated_Code
Includes
MC56F84789_Internal_PFlash_SDM
ProcessorExpert.pe 927 KB Processor E
Project_Headers
Project_Settings
Referenced Projects
SaAnalysispointsManager.apconfig 1 KB
Sources
Events.c 8 KB C Source Fil
Events.h 4 KB C Header Fil
globals.h 4 KB C Header Fil
ProcessorExpert.c 14 KB C Source Fil
DA1.c
ProcessorExpert.c
Cpu.c
Events.c
CAN.c
»2

```
/** ###################################################################
**     Filename    : ProcessorExpert.c
**     Project     : ProcessorExpert
**     Processor   : MC56F84789VLL
**     Version     : Driver 01.14
**     Compiler    : Metrowerks DSP C Compiler
**     Date/Time   : 2012-03-09, 16:14, # CodeGen: 0
**     Abstract    :
**         Main module.
**         This module contains user's application code.
**     Settings    :
**     Contents    :
**         No public methods
**
** ###################################################################*/
/* MODULE ProcessorExpert */


/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
```

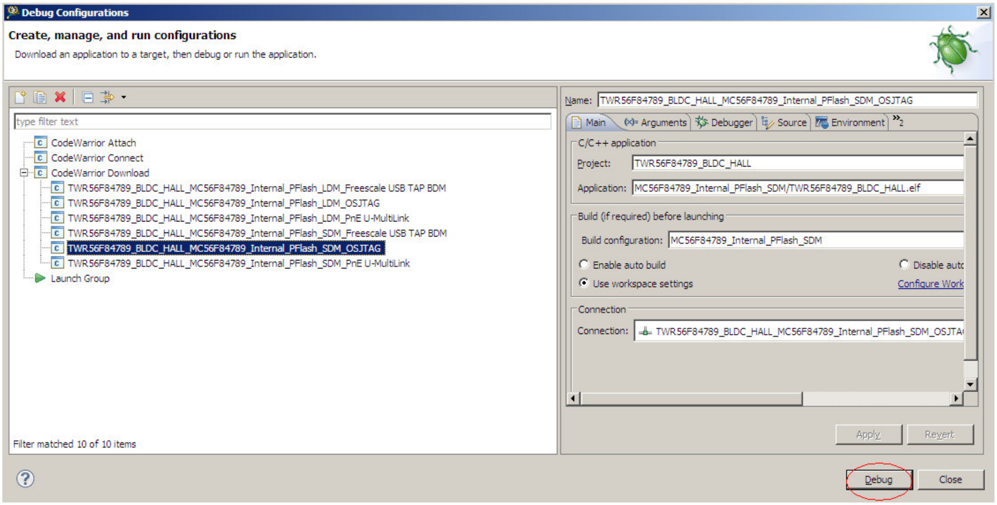
Fig.1 CodeWarrior Project for building

6, When project is built, click the down arrow beside of bug icon (red circle in Fig.2) to pop the pull-down menu of debug, then select “Debug Configurations…” command. In debug configurations interface, select “TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_OSJTAG” option under CodeWarrior Download item, then click “Debug” button (red circle in Fig.3) to download project goal file, then enter debug mode after a few seconds, and stop at the main function entry point. Then either click on a green “Resume” arrow (red circle in Fig.4) once more to start running the application code or use step-by-step debug buttons to trace the application code (shown in Fig.4). Here suggest running the code for following FreeMaster connection.


C/C++ - TWR56F84789_BLDC_HALL/Sources/ProcessorExpert.c - CodeWarrior Development Studio
File Edit Source Refactor Navigate Search Project Run PEMicro Processor Expert Window Help
Debug
C/C++
1 TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_OSJTAG
2 TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_Freescale USB TAP BDM
Debug As
Debug Configurations…
Organize Favorites…
CodeWarrior Projects
Project Panel
File Name
Size
Type
TWR56F84789_BLDC_HALL : MC56F84789
Binaries
Documentation
External_Modules
DA1.c
ProcessorExpert.c

```
/** #################
**     Filename    :
**     Project     :
**     Processor   : MC56F84789VLL
**     Version     : Driver 01.14
**     Compiler    : Metrowerks DSP C Compiler
```

Fig.2 CodeWarrior Project for debug configurations



Create, manage, and run configurations

Download an application to a target, then debug or run the application.

- CodeWarrior Attach
- CodeWarrior Connect
- CodeWarrior Download
  - TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_LDM_Freescale USB TAP BDM
  - TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_LDM_OSJTAG
  - TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_LDM_PnE U-MultiLink
  - TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_Freescale USB TAP BDM
  - TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_OSJTAG
  - TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_PnE U-MultiLink
- Launch Group

Filter matched 10 of 10 items

Name: TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_OSJTAG

C/C++ application

Project: TWR56F84789_BLDC_HALL

Application: MC56F84789_Internal_PFlash_SDM/TWR56F84789_BLDC_HALL.elf

Build (if required) before launching

Build configuration: MC56F84789_Internal_PFlash_SDM

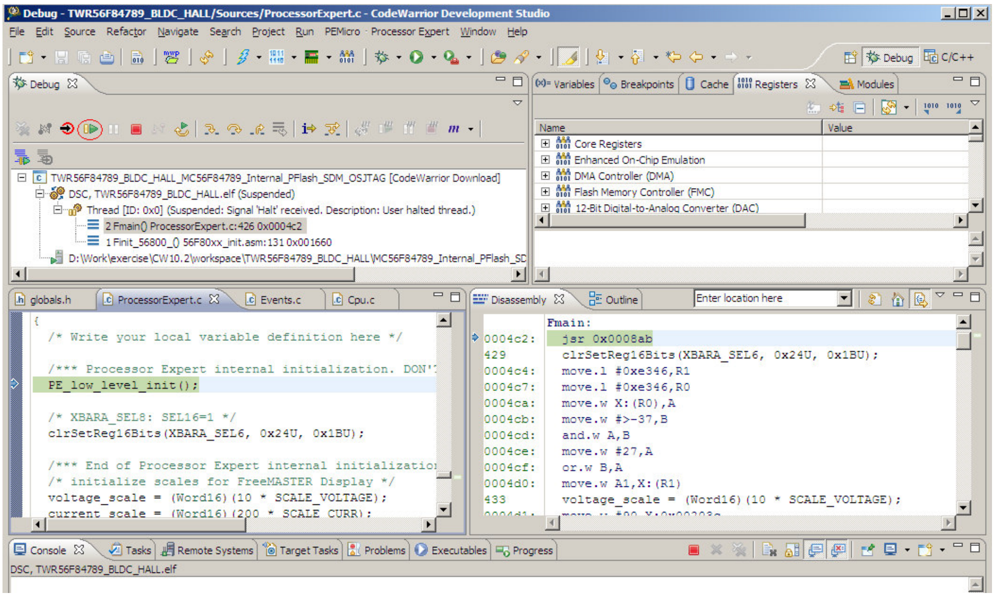
○ Enable auto build ○ Disable auto

● Use workspace settings Configure Work

Connection

Connection: TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_OSJTA

Fig.3 Debug configurations interface



- TWR56F84789_BLDC_HALL_MC56F84789_Internal_PFlash_SDM_OSJTAG [CodeWarrior Download]
  - DSC, TWR56F84789_BLDC_HALL.elf (Suspended)
    - Thread [ID: 0x0] (Suspended: Signal 'Halt' received. Description: User halted thread.)
      - 2 Fmain() ProcessorExpert.c:426 0x0004c2
      - 1 Finit_56800_() 56F80xx_init.asm:131 0x001660
  - D:\Work\exercise\CW10.2\workspace\TWR56F84789_BLDC_HALL\MC56F84789_Internal_PFlash_SD

```
{
  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'
  PE_low_level_init();

  /* XBARA_SEL8: SEL16=1 */
  clrSetReg16Bits(XBARA_SEL6, 0x24U, 0x1BU);

  /*** End of Processor Expert internal initializatio
  /* initialize scales for FreeMASTER Display */
  voltage_scale = (Word16)(10 * SCALE_VOLTAGE);
  current_scale = (Word16)(200 * SCALE_CURR);
```

```
          Fmain:
0004c2:     jsr 0x0008ab
429         clrSetReg16Bits(XBARA_SEL6, 0x24U, 0x1BU);
0004c4:     move.l #0xe346,R1
0004c7:     move.l #0xe346,R0
0004ca:     move.w X:(R0),A
0004cb:     move.w #>-37,B
0004cd:     and.w A,B
0004ce:     move.w #27,A
0004cf:     or.w B,A
0004d0:     move.w A1,X:(R1)
433         voltage_scale = (Word16)(10 * SCALE_VOLTAGE);
```

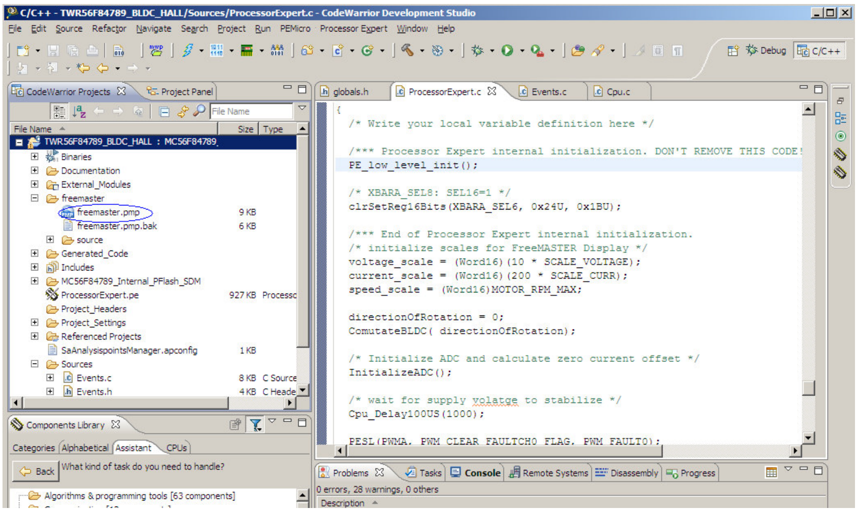
DSC, TWR56F84789_BLDC_HALL.elf

Debug - TWR56F84789_BLDC_HALL/Sources/ProcessorExpert.c - CodeWarrior Development Studio
File Edit Source Refactor Navigate Search Project Run PEMicro Processor Expert Window Help
Debug
C/C++
Variables
Breakpoints
Cache
Registers
Modules
Name
Value
Core Registers
Enhanced On-Chip Emulation
DMA Controller (DMA)
Flash Memory Controller (FMC)
12-Bit Digital-to-Analog Converter (DAC)
globals.h
ProcessorExpert.c
Events.c
Cpu.c
Disassembly
Outline
Enter location here
Console
Tasks
Remote Systems
Target Tasks
Problems
Executables
Progress

Fig.4 CodeWarrior Project for debug mode

2.5 Setup Connection between FreeMaster and Target

1, Open FreeMaster project of “BLDC Motor Control Demo with Hall Sensor” by double-clicking on freemaster.pmp in freemaster folder of “CodeWarrior Projects” tab of the CodeWarrior project window (blue circle in Fig.5). Alternatively the FreeMaster project file can be also opened directly from {Project}\freemaster\freemaster.pmp.



```
{
  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!
  PE_low_level_init();

  /* XBARA_SEL8: SEL16=1 */
  clrSetReg16Bits(XBARA_SEL6, 0x24U, 0x1BU);

  /*** End of Processor Expert internal initialization.
  /* initialize scales for FreeMASTER Display */
  voltage_scale = (Word16)(10 * SCALE_VOLTAGE);
  current_scale = (Word16)(200 * SCALE_CURR);
  speed_scale = (Word16)MOTOR_RPM_MAX;

  directionOfRotation = 0;
  ComutateBLDC( directionOfRotation);

  /* Initialize ADC and calculate zero current offset */
  InitializeADC();

  /* wait for supply volatge to stabilize */
  Cpu_Delay100US(1000);

  PESL(PWMA, PWM_CLEAR_FAULTCH0_FLAG, PWM_FAULT0);
```

Fig.5 Launch FreeMaster Project

2, In order to enable FreeMaster communication, MC9S08JM60 MCU has to be configured to operate as SCI to USB bridge.

A virtual COM port number can be identified in MS Windows Device Manager, which is labeled with “OSBDM/OSJTAG – CDC Serial Port (http://www.pemicro.com/osbdm)”. Remember this port number to be used in FreeMaster communication setting options.

Note: the virtual COM port number can be different with the one (the virtual port number is COM19) in Fig.6.

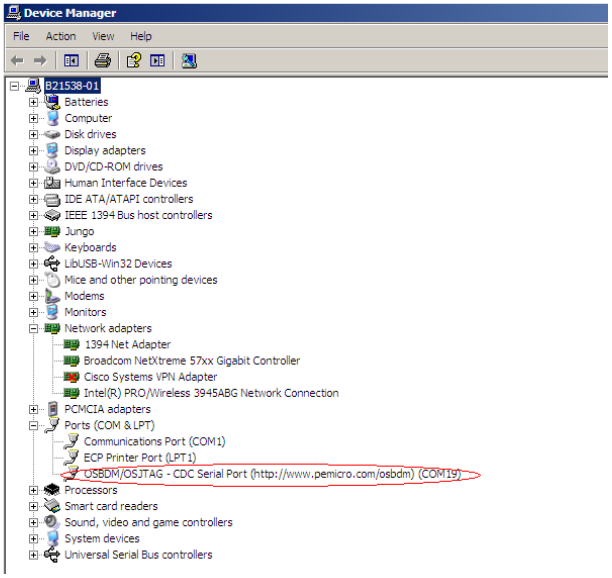
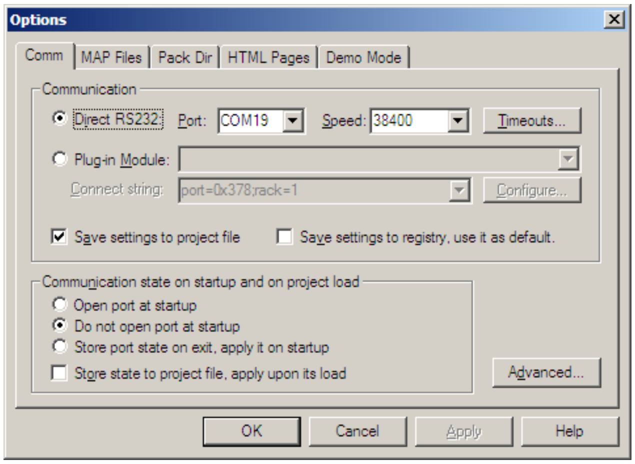

Device Manager
File Action View Help
B21538-01
Batteries
Computer
Disk drives
Display adapters
DVD/CD-ROM drives
Human Interface Devices
IDE ATA/ATAPI controllers
IEEE 1394 Bus host controllers
Jungo
Keyboards
LibUSB-Win32 Devices
Mice and other pointing devices
Modems
Monitors
Network adapters
1394 Net Adapter
Broadcom NetXtreme 57xx Gigabit Controller
Cisco Systems VPN Adapter
Intel(R) PRO/Wireless 3945ABG Network Connection
PCMCIA adapters
Ports (COM & LPT)
Communications Port (COM1)
ECP Printer Port (LPT1)
OSBDM/OSJTAG - CDC Serial Port (http://www.pemicro.com/osbdm) (COM19)
Processors
Smart card readers
Sound, video and game controllers
System devices
Universal Serial Bus controllers

Fig.6 OSJTAG Virtual COM Port

3, Configure FreeMaster options correctly. Click “Options…” command in “Project” menu, then select “Comm” tab in “Options” interface. Direct RS232 has to be selected for FreeMaster communication with proper COM port (the port number is COM19 in this example, which can be gotten from step 8) and the speed (baud rate) of 38400, finally click “OK” button to complete configuration (shown in Fig.7).



Communication

● Direct RS232 Port: COM19 Speed: 38400

○ Plug-in Module:

Connect string: port=0x378;rack=1

☑ Save settings to project file ☐ Save settings to registry, use it as default.

Communication state on startup and on project load

○ Open port at startup

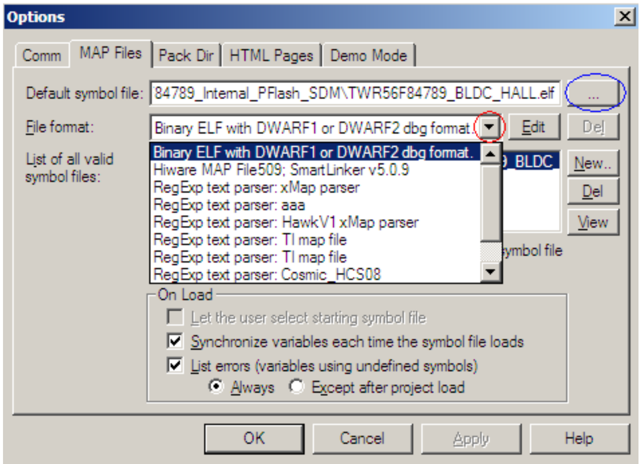
● Do not open port at startup

○ Store port state on exit, apply it on startup

☐ Store state to project file, apply upon its load

Fig.7 FreeMaster Communication Setting Options

If required, please select “MAP Files” tab in “Options” interface. Open demo symbol file by clicking “open” button (blue circle in Fig.8), then selecting symbol file of TWR56F84789_BLDC_HALL.elf from the folder of {Project}\MC56F84789_Internal_PFlash_SDM, and select correct file format of “Binary ELF with DWARF1 or DWARF2 dbg format.” from pull-down menu by clicking the button (red cycle in Fig.8) with down arrow in the right side of text box of “File format” item, finally click “OK” button to complete configuration (shown in Fig.8).



Default symbol file: 84789_Internal_PFlash_SDM\TWR56F84789_BLDC_HALL.elf

File format: Binary ELF with DWARF1 or DWARF2 dbg format.

List of all valid symbol files:

Binary ELF with DWARF1 or DWARF2 dbg format.
Hiware MAP File509; SmartLinker v5.0.9
RegExp text parser: xMap parser
RegExp text parser: aaa
RegExp text parser: HawkV1 xMap parser
RegExp text parser: TI map file
RegExp text parser: TI map file
RegExp text parser: Cosmic_HCS08

On Load

☐ Let the user select starting symbol file

☑ Synchronize variables each time the symbol file loads

☑ List errors (variables using undefined symbols)

● Always ○ Except after project load

Fig.8 FreeMaster MAP Files Setting Options

4, Once the FreeMaster options are configured, FreeMASTER communication can be started. To start/stop the communication by clicking the red “STOP” button in the button toolbar of the FreeMASTER window (shown in Fig.9).

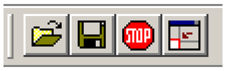
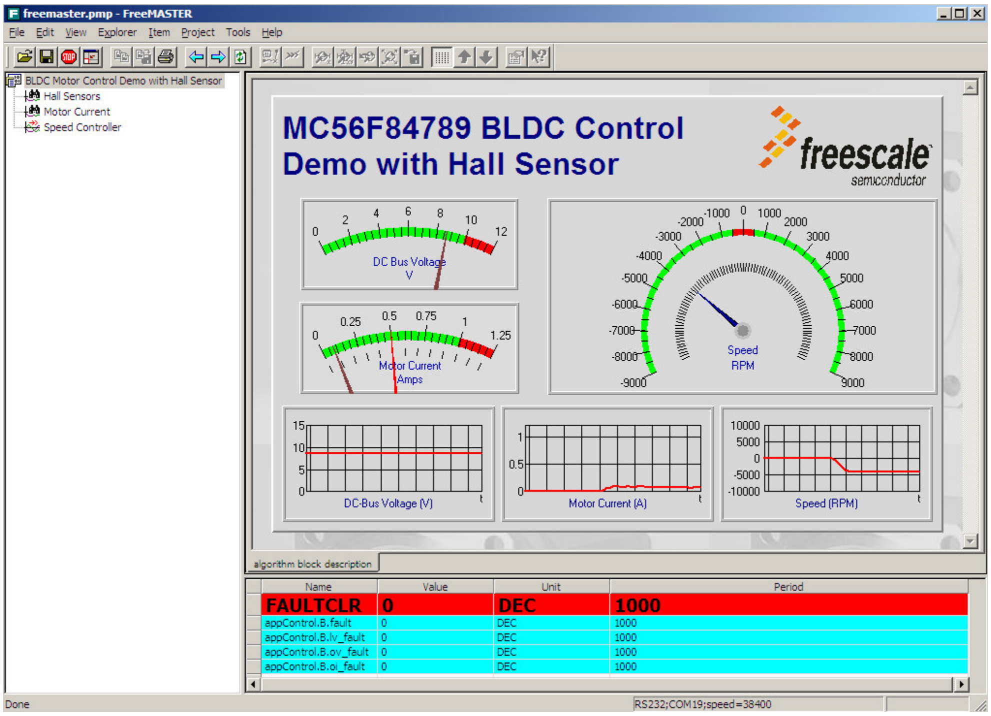


Fig.9 FreeMaster Button Toolbar

3, Control Features of FreeMaster Project

The FreeMaster control page of the BLDC Motor Control Demo with Hall Sensor is shown in Figure 10, which offers following features:

- Three gauges display motor speed, DC-Bus voltage and motor current
- Three strip graphs to plot motor speed, DC-Bus voltage and motor current
- Fault status window, which displays actual application faults status
- Fault clear variable to clear/acknowledge detected application faults
- Set required motor speed by clicking on speed gauge (by default set to 0RPM)
- Set motor current limit level by clicking on motor current gauge (by default set to 0.5Amps)

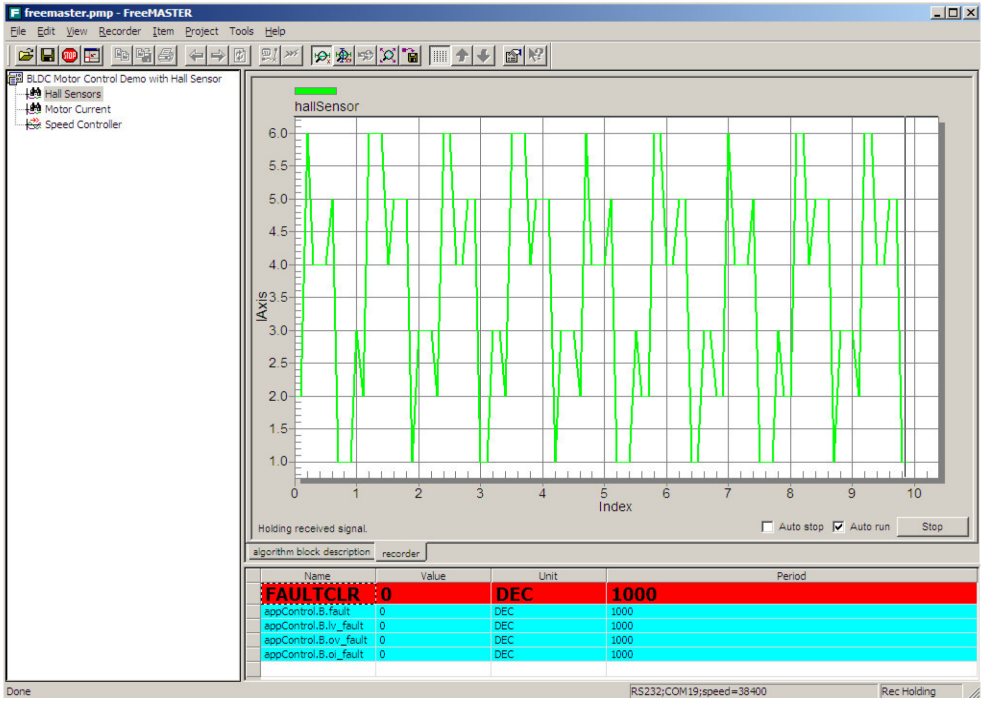


Name	Value	Unit	Period
FAULTCLR	**0**	**DEC**	**1000**
appControl.B.fault	0	DEC	1000
appControl.B.lv_fault	0	DEC	1000
appControl.B.ov_fault	0	DEC	1000
appControl.B.oi_fault	0	DEC	1000

Fig.10 FreeMaster Control Page

In FreeMaster project, two Recorder windows and one Scope window can be open. To open Recorder or Scope window, select one from a list on left project-tree window (shown in Fig.10). The Recorder windows will enable you to watch hall sensor value (shown in Fig.11) and motor current (shown in Fig.12). The Scope window will allow you to watch required and actual motor speed and PWM duty-cycle (shown in Fig.13).

Note: please refer to FreeMaster User Manual for a complete list of features.

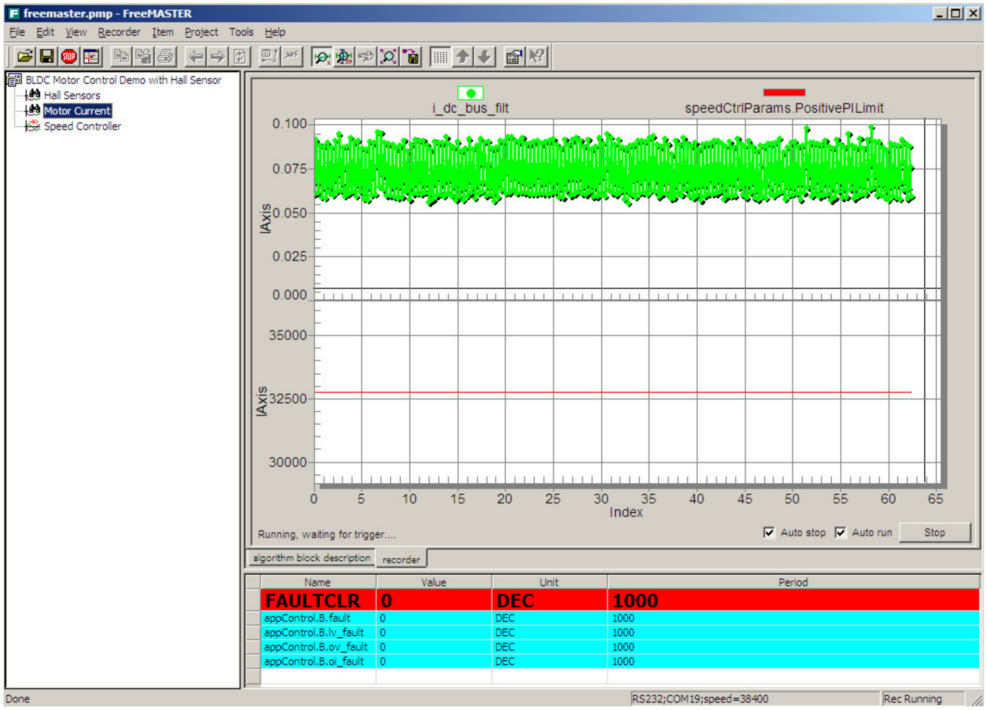


Holding received signal.

Name	Value	Unit	Period
FAULTCLR	0	DEC	1000
appControl.B.fault	0	DEC	1000
appControl.B.lv_fault	0	DEC	1000
appControl.B.ov_fault	0	DEC	1000
appControl.B.oi_fault	0	DEC	1000

Done RS232;COM19;speed=38400 Rec Holding

Fig.11 FreeMaster Recorder Screenshot – Hall Sensors

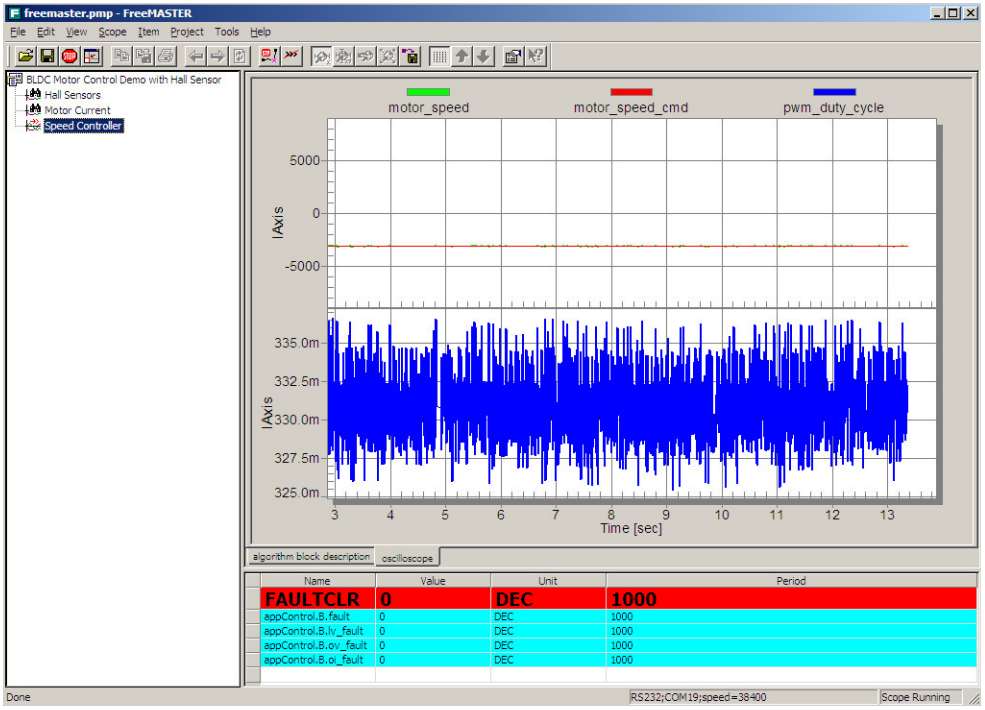


Running, waiting for trigger....

Name	Value	Unit	Period
FAULTCLR	0	DEC	1000
appControl.B.fault	0	DEC	1000
appControl.B.lv_fault	0	DEC	1000
appControl.B.ov_fault	0	DEC	1000
appControl.B.oi_fault	0	DEC	1000

Done RS232;COM19;speed=38400 Rec Running

Fig.12 FreeMaster Recorder Screenshot – Motor Current



Name	Value	Unit	Period
FAULTCLR	**0**	**DEC**	**1000**
appControl.B.fault	0	DEC	1000
appControl.B.lv_fault	0	DEC	1000
appControl.B.ov_fault	0	DEC	1000
appControl.B.oi_fault	0	DEC	1000

Fig.13 FreeMaster Scope Screenshot – Speed Controller