ON Semiconductor

Is Now

onsemi™

To learn more about onsemi™, please visit our website at
www.onsemi.com

AND9369/D

AX MicroLAB Software Manual

ON Semiconductor®

www.onsemi.com

APPLICATION NOTE

Introduction

The AX MicroLAB is a utility tool that can be used to easily configure ON Semiconductor microcontrollers.

The main features of the AX MicroLAB are:

- Hardware and Functions Configuration through Graphic Interface
- Preset for ON Semiconductor DVK−2
- Built−in Code Examples

The AX MicroLAB generates a code for the given settings that automatically sets the microcontroller registers.

AX MicroLAB

Starting a New Project

To create a new project in the AX MicroLAB run the program and chose the option “New Project” on the dialog window. A new panel will be shown where the project directory and the microcontroller can be chosen. Once both the path and the desired microcontroller have been set, click on the “Ok” button. The program will generate an “.xml” file, source codes and headers inside the chosen path.

Opening an Existing Project

To open an existing project run the program and chose “Open Existing Project”. Select the directory where the “.xml” savefile is located. The selected project will be loaded.

Note that only a project can be open at a time.

The Main Panel

The AX MicroLAB main panel is shown in Figure 1. It is divided in four areas: pin configuration (red), hardware configuration (green), function configuration (blue) and code generation (violet).

In the pin configuration area it is possible to select the function for each pin. Almost every pin can be used as a GPIO (General Purpose I/O) pin. If a pin is set as GPI, it is possible to configure the pull−up resistor function if needed. If a pin is set as GPO, it is possible to configure whether the static output should be 1 or 0. It is also possible to configure each pin with the associated alternate functions. The complete list of alternate functions for each pin can be found in the AX8052 Family Programming Manual.

February, 2016 - Rev. 2

Publication Order Number:
AND9369/D

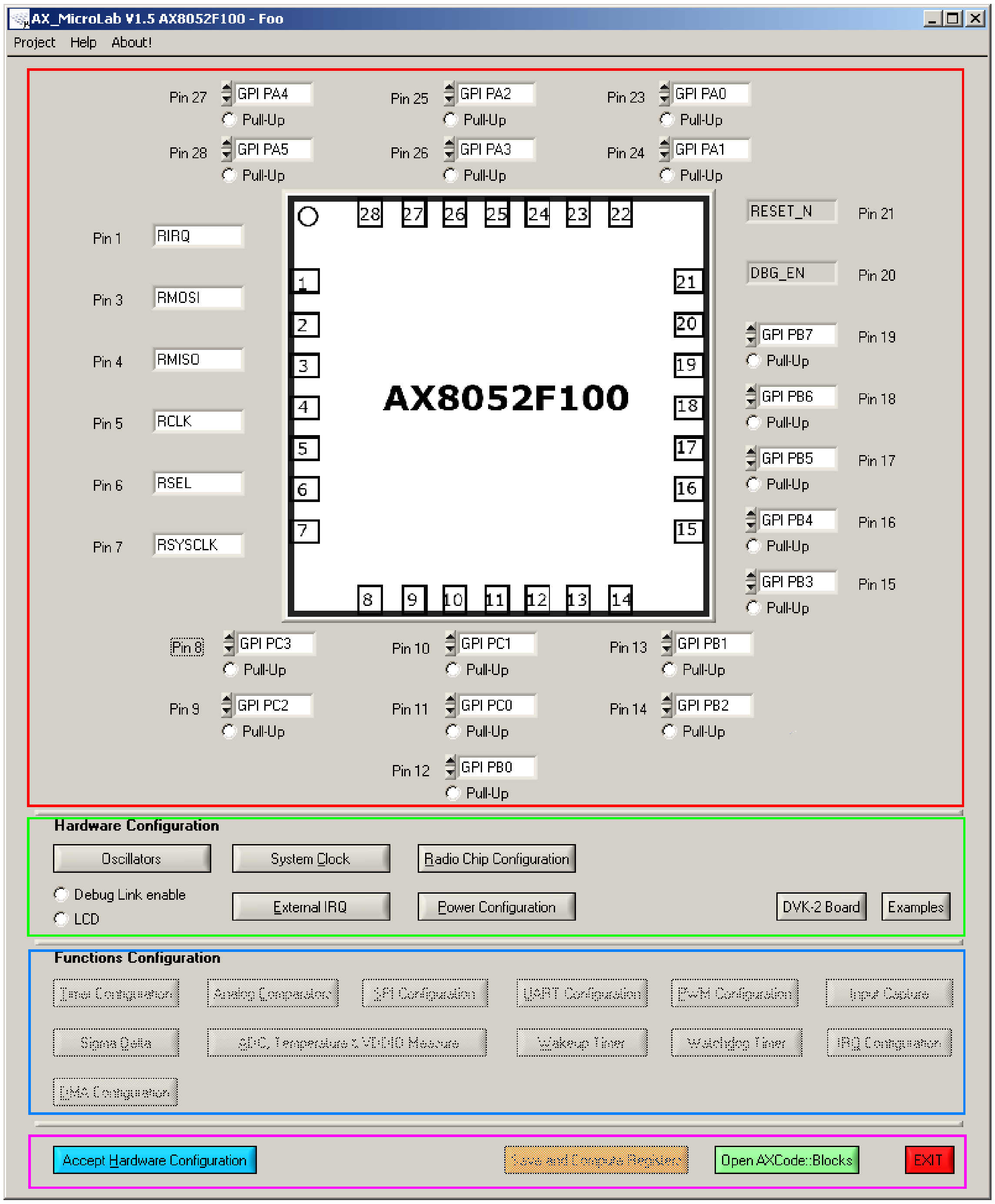

Figure 1. AX MicroLAB Main Panel

The hardware configuration area allows to set other hardware settings such as the System Clock or the Radio Chip Configuration. It is here possible to enable the Debug Interface used with the AXSDB. If an ON Semiconductor DVK−2 Mainboard is used for the project the “DVK−2 Board” option can be activated. This option provides a standard pin configuration, that can also be modified. If an LCD is connected at the SPI ports of the microcontroller, the “LCD” button can be activated. This option allows an easy configuration of the LCD thanks to the respective functions in the Libmf library. It is also possible to load some example preset. Each option in this area can interact with the pin configuration changing and/or blocking some pin setting.

Once the hardware configuration has been completed it is possible to click on “Accept Hardware Configuration” to access the Function configuration area. Here the different chip functions can be configured. The options in this area can not modify the pin configuration and it is possible that some functions can not be enabled for the given hardware configuration. In this case it is possible to modify the actual hardware setting by clicking on “Change Hardware Configuration”.

Finally the “Save and Compute Registers” button will save the project and generate a code that can be downloaded onto the microcontroller using the AXCode::Blocks editor which can be opened by clicking on “Open Project Editor” (AX8052−IDE Software Package must be installed).

It is recommended to proceed as follows:

1. Chose the hardware settings. Hardware configuration panels automatically set the needed pins for the entered configuration. This is an easy way to configure most of the pins.
2. Set the pins in the desired configuration. Error messages will appear in case the entered pin configuration is wrong.
3. Click on the “Accept Hardware Configuration” button.
4. Proceed with the function configuration. Functions may be unavailable according to how the hardware has been configured. To go back to the hardware configuration click on the “Change Hardware Configuration” button.
5. Click on the “Save and Compute Registers” button. This writes the firmware.
6. Click on “Open Project Editor” to start the AXCode::Blocks IDE.

Hardware Configuration

Please keep the AX8052 Family Programming Manual as reference for the settings here described.

Oscillators

This panel allows to declare the oscillators that are used in the project and their respective operating frequencies. The declared oscillator will be available in other panels as clock source. Activating an oscillator will automatically set the corresponding pins in the pin configuration.

It is also possible to access the calibration panel for the Low Power Oscillator (LPOSC) and for the Fast RC Oscillator (FRCOSC).

LPOSC Calibration

To enable the LPOSC calibration simply chose a calibration source from the menu. The optimal prescaler and the maximal calibration filter constant k_{filt} are calculated automatically. It is possible to set the desired k_{filt} as a percentage of the maximal k_{filt}.

For more details on the LPOSC calibration, please consult the AX8052 Family Programming Manual.

FRCOSC Calibration

To enable the FRCOSC calibration simply chose a calibration source and a prescaler. The resulting frequency is displayed. Keep in mind that it is desirable to keep the resulting frequency between 500 and 1000 Hz. The maximal calibration filter constant k_{filt} is calculated automatically. It is possible to set the desired k_{filt} as a percentage of the maximal k_{filt}.

For more details on the FRCOSC calibration, please consult the AX8052 Family Programming Manual.

System Clock

Here the system clock source can be chosen amongst the oscillators activated in the “Oscillators” panel. The source can be scaled, the resulting frequency is displayed.

Radio Chip Configuration

If the chosen microcontroller is the AX8052F100 it will be possible to load preset configurations for different ON Semiconductor radio chips. Other radio chips can be used and manually configured. If the “None” option is selected in the “Radio Chip” menu, the pins usually used for the radio chip can be configured as GPIO.

If other microcontrollers are chosen the corresponding radio chip will be automatically set with the appropriate preset configuration.

If the DVK−2 mainboard is used together with add−on modules then the appropriate radio chip must be selected here.

External IRQ

The external interrupt request can be enabled here. If an external interrupt source is activated the selected pin will be set accordingly. To set the priority of the External interrupt go to the “IRQ Configuration” panel in the Function Configuration area.

Power Configuration

When the microcontroller goes in sleep mode the two XRAM memory block can be retained. In this panel it is possible to chose which memory block to retain.

Debug Link Enable

By choosing this options the needed pin will be automatically configured and the generated code will initialize the Debug Link Interface.

LCD

This option can be activated if an LCD is connected to the microcontroller through the SPI ports. The generated code will initialise the LCD with the appropriate Libmf functions.

DVK−2 Board

This button sets a standard hardware configuration for the ON Semiconductor DVK−2 Mainboard. This template is a suggested configuration to let the ON Semiconductor DVK−2 Mainboard work properly. By chosing it, it will still be possible to modify the pin configuration and the hardware settings.

Examples

The AX MicroLAB provides some example presets. Each preset can modify the pin configuration and configure some of the chip functions. These examples are intended for the

ON Semiconductor DVK−2 Mainboard, although they can be downloaded on any configurable ON Semiconductor microcontroller. If you are using an ON Semiconductor DVK−2 Mainboard, it is preferable to first click on the "DVK−2 Board" button, then chose the desired example. Some of the provided examples can run at the same time.

The available presets in AX MicroLAB are:

- Sigma−Delta DAC: a DMA buffer containing a sine signal is created. The first DMA channel is set to work with Timer 0 configured as ΣΔ converter. The output signal can be measured with a scope on pin PA0 (pin 23 for the AX8052F100, pin 30 on AX8052F151 and AX8052F143). This example illustrates the initialisation and the use of the DMA and the ΣΔ functionality of the timers.
- ADC Temperature/VDDIO: the ADC is configured to sample the temperature by using the microcontroller built−in temperature sensor and the VDDIO. The measured temperature and VDDIO are written on a DMA buffer. An interrupt call is generated when the DMA buffer is full. The mean value of the temperatures and of the VDDIO written in the buffer will be displayed on the Debug Link window and on the display in case a DVK−2 Board is used. This example illustrates the working principle of the ADC together with DMA interrupts handling.
- Stopwatch: Timer 2 is configured as a reference for a stopwatch that can be controlled with SW4 and SW5 buttons on the DVK−2 Board. The time will be displayed on the Debug Link window and on the display. This example illustrates GPIO interrupts handling and how to use timers for timing reference.
- Frequency Measure: Input capture is used to determine the frequency of a signal. Timer 1 is used as the input capture source while Timer 2 is used to generate a signal that triggers the input capture. Timer 2 can be modified to generate different frequencies. This example allows to measure frequencies between 306 Hz and 1.25 MHz. This example shows how to use the input capture function combined with the DMA.
- Standby Mode: this preset demonstrates how to correctly put the microcontroller in standby mode and how to wake it up using a GPIO interrupt (SW5 on the DVK−2 Board).
- Wakeup Timer: this example shows how it is possible to wake up the microcontroller from sleep mode using a wakeup timer.

Function Configuration

Please keep the AX8052 Family Programming Manual as reference for the settings here described.

Timer Configuration

The three timers can be configured in this windows. Timers are used by other functions such as UART, PWM (Output Compare), Input Capture and ΣΔ.. The Timer operating mode and period can be set, the resulting timer frequency is displayed. In the timer configuration the interrupt mode can be chosen for each timer. In order to enable the interrupt for each timer it is necessary to activate the corresponding cell in the "IRQ Configuration" panel.

Analog Comparators

In order to enable this function the correct pin configuration must be set. The input an the reference for the two analog comparators can be chosen. The result of the comparation is given on the respective pins configured ad "COMPO".

SPI Configuration

In order to enable this function the correct pin configuration must be set. The microcontroller can be configured as Master or Slave device. If the microcontroller has to be operated as slave peripheral, the three−wires or the four−wires configuration can be chosen.

UART Configuration

In order to enable this function the correct pin configuration must be set. Enabling one of the two UART interface will automatically enable the selected timer. The selected timer is reserved for the UART interface and other function can not use it as reference. The two UART interfaces can use the same timer if it is desired that they have the same baud rate.

PWM Configuration

In order to enable this function the correct pin configuration must be set. Once one of the PWM is enabled the corresponding timer is activated, if it is not already on, in the Timer Configuration Windows where it can be configured. This function will not reserve the timers, hence other functions can use the same timer as the PWM.

Input Capture

This window allows to fully configure both Input Capture channels. Once one of the Input Capture channel is enabled the selected timer is activated, if it is not already on, in the Timer Configuration Windows where it can be configured. This function will not reserve the timers, hence other functions can use the same timer as the Input Capture.

Sigma Delta

In this window the ΣΔ feature of the microcontroller can be configured. To each Timer correspond a ΣΔ channel. If one of this channel is activated the corresponding timer will be enabled and reserved in the Timer Configuration window. Therefore other function will not be able to use a timer set to ΣΔ.

ADC, Temperature & VDDIO Measure

Here the four ADC channels can be enabled and configured. Further it is possible to set the conversion control, the power saving mode and the clock source for the ADC.

Wakeup Timer

The two wakeup timers of the microcontroller can be activated and configured.

Watchdog Timer

The watchdog timer can be activated and configured.

IRQ Configuration

Interrupt sources can be activated in this panel and their respective priority can be set. If an interrupt source is activated, the generated code will define an Interrupt Service Routine (ISR) and an interrupt flag. If the GPIO interrupt source is activated it will be possible to configure the interrupt on port change for all the pins of the microcontroller.

DMA Configuration

The two DMA channels can be configured. The DMA buffer can be initialised using the prototype contained in the file "easy_dma.h"

Generated Files

When a new project is created the following files are generated in the project folder:

File	Description	Overwritten on "Save and Compute Registers"
..\save.xml	.xml file containing the project informations	yes
..\AXML.cbp	Codeblocks project file	no
..\AXML.layout	Codeblocks projet layout	no
..\irq.h	Header file containing the prototype for the activated ISR and interrupt flags	yes
..\definitions.h	Header file describing special functions and presets activated for the project	yes
..\xy_example.c	Source code containing the code for the corresponding example preset	no
..\xy_example.h	Header file for xy_example.c	no
..\main.c	Main code	no
..\setRegisters.c	Source code containing functions for the configuration of the microcontroller with the selected settings.	yes
..\setRegisters.h	Header file for setRegisters.c	no
..\easy_dma.c	Definition of the functions declared in easy_dma.h	no
..\easy_dma.h	Prototypes of utility functions and buffer descriptor for the initialisation of DMA channels	no
..\utility.c	Definition of the functions declared in utility.h	no
..\utility.h	Prototypes of utility function used in some example	no

Templates

Hardware and functions configurations can be saved as template for other projects.

In the menu bar click on "Project" and then on "Save as Template". The actual project configuration will be saved.

To load a template click on "Project" and the go under "Load Template". By choosing one of the available templates the corresponding configuration will be loaded.

The templates are stored in the "Templates" folder in the program install directory. To erase a template simply delete the template file.

Templates can be shared between projects with different microcontrollers.

Firmware Structure

Main.C

This file represent the skeleton of the firmware. The main file has the simplified dummy structure shown in Code 1.

Code 1.
Dummy Structure of the Main Function of the Firmware

```
1     void main(void)
2     {
3         _sdcc_external_startup();
4
5         ax8052_setup();
6
7     //Examples Initialization
8         EA = 1;
9         Ex1_init();
10        Ex2_init();
11        Ex3_init();
12        ...
13        Exn_init();
14    //Main Loop
15        for (;;)
16        {
17            EA = 0;
18            if(Ex1_work())
19                continue;
20            if(Ex2_work())
21                continue;
22            if(Ex3_work())
23                continue;
24            ...
25            if(Exn_work())
26                continue;
27            enter_standby();
28            EA = 1;
29        }
30    }
```

Lines 3 and 5 are responsible for the initialisation of the microcontroller: the function "_sdcc_external_startup" calls the function "setRegisters" that is defined together with "ax8052_setup" in the "setRegisters.c" file. These two functions configure the registers of the microcontroller as previously defined in the AX_MicroLAB.

Lines 9 to 13 consist of the initialisation of the examples. Each example provided with the AX_MicroLAB has an init−function which should be called at this point.

Lines 15 to 29 constitute the main loop. This infinite loop contains an if−statement for each example. Each example provided with the AX_MicroLAB has a work−function which should be called here. If an example work−function has a task to execute, it does it and gives 1 as return value. In this case the infinite loop would start again. If any of the work−function has nothing to do, they will all return 0 as result. In this case the program execution will reach line 27 where the microcontroller is put in standby mode until it is waken again, for instance by an interrupt call.

Example Files

Example files have the simplified structure shown in Code 2.

To each example source code is paired an header file with the prototypes of the functions and the declaration of the variables used by the example.

Lines 4 to 6 consist in the variables definitions.

Line 9 to 13 shows a dummy initialization function. This function can have arguments and does not need a return value, although a return value can be useful to implement an initialization check (for instance: return 1 if the example has been correctly initialized).

Code 2. Dummy Structure of an Example File

```
1   #include "xy_example.h"
2
3   #ifdef XY
4   uint16_t a;
5   uint16_t b;
6   uint16_t c;
7   #endif
8
9   void xy_init(void){
10  #ifdef XY
11      various initialisation instruction...
12  #endif
13  }
14
15  uint8_t xy_work(void){
16  #ifdef XY
17      if(event)
18      {
19          various instruction...
20          return 1;
21      }
22  #endif
23      return 0;
24  }
```

Line 15 to 24 constitute the work−function. This function contains an if−statement. If some event occours (for instance a flag is activated by an interrupt) then some instructions are executed and the value 1 is returned. Otherwise the returned value will be 0.

It is important to note the use of the compiler instructions "ifdef" and "endif". The "definitions.h" files contains the definitions for the activated examples. In the considered case, if the "definitions.h" file contains the definition "#define XY" then the compiler will consider the code between "ifdef" and "endif". Otherwise this code will not be compiled and the example would simply be deactivated.

It is preferable to build the firmware using the shown structure. If the microcontroller has to execute numerous tasks, each task can be programmed with an init− and a work−function. This method allows more tasks to run together in the same firmware.

Easy DMA

Each project generated with the AX_MicroLAB is provided with an “easy_dma” source code and header. These two files contain some utility for the initialisation of the two DMA channels.

In Code 3. we can see the declaration of the structure “DMA_descriptor”.

Code 3. DMA Buffer Descriptor Structure

```
struct DMA_descriptor{

    void __xdata *bufaddr;

    uint16_t buflen;

    uint16_t actlen;

    struct DMA_descriptor __xdata *bdaddr;

};
```

“bufaddr” is a pointer to a buffer saved in the XRAM memory of the microcontroller. The buffer can be initialized as an array in the XRAM. In this case the first element of the buffer descriptor should be the pointer to this array.

“buflen” is the length of the buffer. Again, if the buffer is initialized as an array in the XRAM, “buflen” is the length of the array.

“actlen” represents the actual length of the buffer. This value is written by the DMA once the buffer has been filled.

“bdaddr” is the pointer to the next buffer descriptor. If the pointer has value 0xffxx the DMA will stop at the end of the current buffer.

The “easy_dma” files also implement the functions “start_DMA0” and “start_DMA1”. These functions require a pointer to a buffer descriptor as argument and can be used to start the two DMA channels.

PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:
Literature Distribution Center for ON Semiconductor
19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA
Phone: 303−675−2175 or 800−344−3860 Toll Free USA/Canada
Fax: 303−675−2176 or 800−344−3867 Toll Free USA/Canada
Email: orderlit@onsemi.com

N. American Technical Support: 800−282−9855 Toll Free USA/Canada
Europe, Middle East and Africa Technical Support:
Phone: 421 33 790 2910
Japan Customer Focus Center
Phone: 81−3−5817−1050

ON Semiconductor Website: **www.onsemi.com**

Order Literature: http://www.onsemi.com/orderlit

For additional information, please contact your local Sales Representative