BL 2600

CONTENTS

- Objective
- Methodology
- Requirements
- Project Breakup
- Approaches
- Introduction
- Subsystems in BL2600
- The G.U.I
- Backend of The GUI
- Conclusion

OBJECTIVE

- To design a Graphic User Interface which enables the user to modify the configurable I/O, A/D and D/A subsystems present on the BL2600 board, as per the user's requirement.

METHODOLOGY

- Comprised initially of study of the documentation of the BL2600 board and Dynamic C.
- The GUI was designed using VC++/VB.
- The backend programming is done in Dynamic C along with the linking.

REQUIREMENT

- BL2600 kit
- Dynamic C
- Microsoft Visual Studio

Project Breakup

- Phase I : Feasibility Presentation

  I (a) : study of all the available theory and documentation required for the completion of project.
  I (b) : implementation of the assignment involving the control of the L.E.D status from the designed webpage on the RCM 3400 board.

- Phase II : Configuring the available ports/pins for making BL2600 for “Generic Use”.

- Phase III : Testing the implemented procedure.

Approaches

Alternative approaches :

1. JAVA
2. VC++/VB

INTRODUCTION

- The BL2600 is a high-performance, C-programmable single-board computer .
- Incorporates a Rabbit3000 microprocessor with a speed of 44.2MHz.
- The I/O can be expanded with RabbitNet peripheral cards.

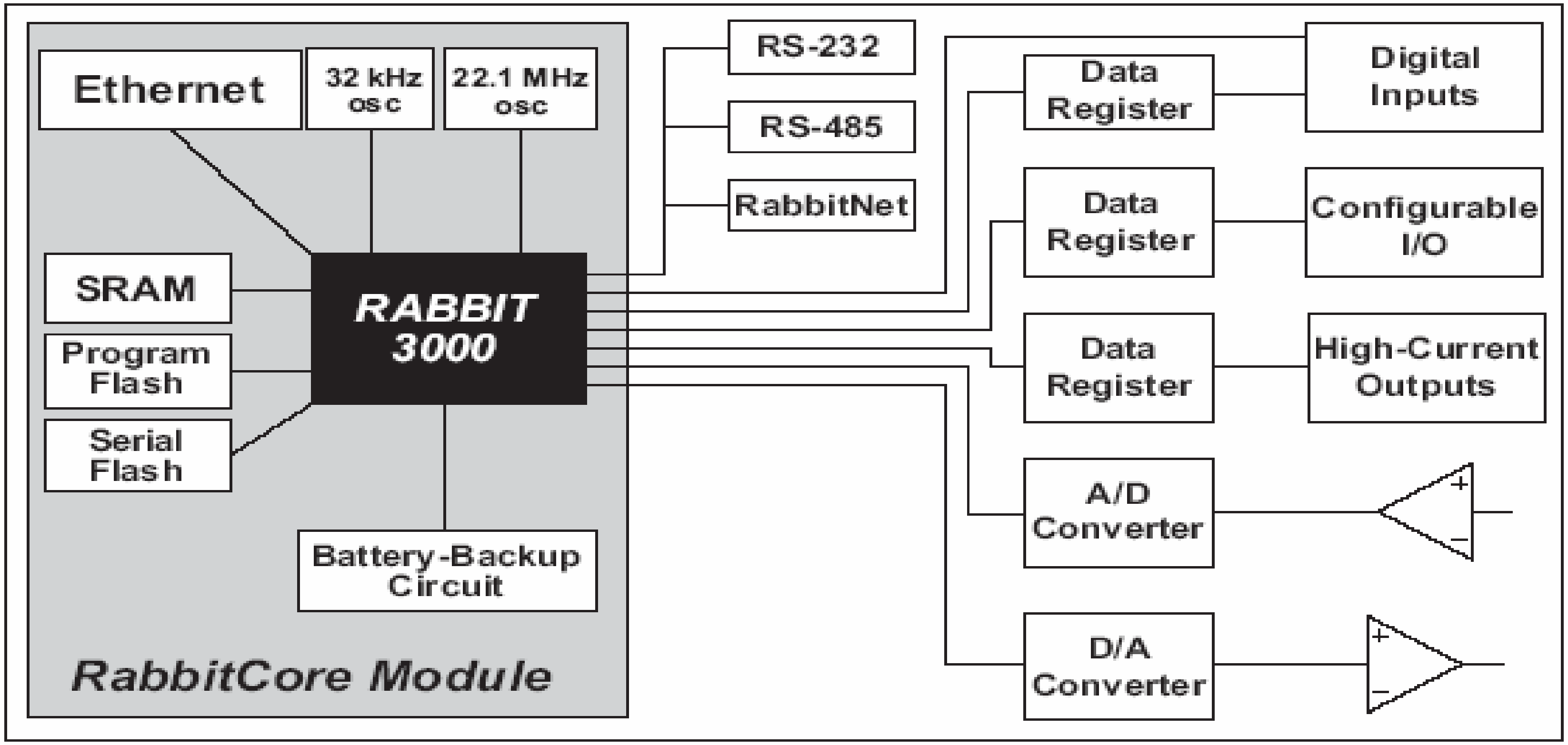
Contd.

- 512K static RAM and 512K flash memory standard.
- 36 digital I/O .
- 12 analog channels .
- Three serial ports (2 RS-232 or 1 RS-232 with RTS/CTS, 1 RS-485 or RS-232).
- Two Rabbit Net expansion ports

Subsystems integrated in BL 2600 :

- Digital I/O
- Serial Communication
- Memory
- A/D Converter Inputs
- D/A Converter Outputs

Subsystems integrated in BL 2600



Digital Input/Output:

- Digital Inputs
- High Current Digital Outputs
- Configurable I/O

Configurable I/O

(DIO 00-DIO 15)

- The BL2600 has 16 configurable I/O that may be configured individually in software as either digital inputs or as digital outputs.
- By default, a configurable I/O channel is a digital input, but may be set as a digital output by using the **digOutConfig** function call.
- DIO 00 – DIO15 are configured as inputs when their status is “0” and configured as outputs if “1”, all the bits can be set or cleared individually.

High-Current Digital Outputs

•The BL2600 has four high-current digital outputs, HOUT0–HOUT3, which can each sink or source up to 2 A .

•When the BL2600 is first powered up or reset, all the outputs are disabled, that is, at a high-impedance tristate, until the **digHoutConfig** software function call is made. The **digHoutConfig** call sets the initial state of each high-current output according to the configuration specified by the user, and enables the digital outputs to their initial status.

.

Analog to Digital Converter

- The single A/D converter chip used in the BL2600 has a resolution of 12 bits (11 bits for the value and one bit for the polarity).
- The A/D converter chip has a programmable amplifier and each external input has circuitry that provides scaling and filtering.
- The A/D converter chip can only accept positive voltages and adjacent A/D converter inputs are paired to make bipolar measurements.

Digital to Analog Converter

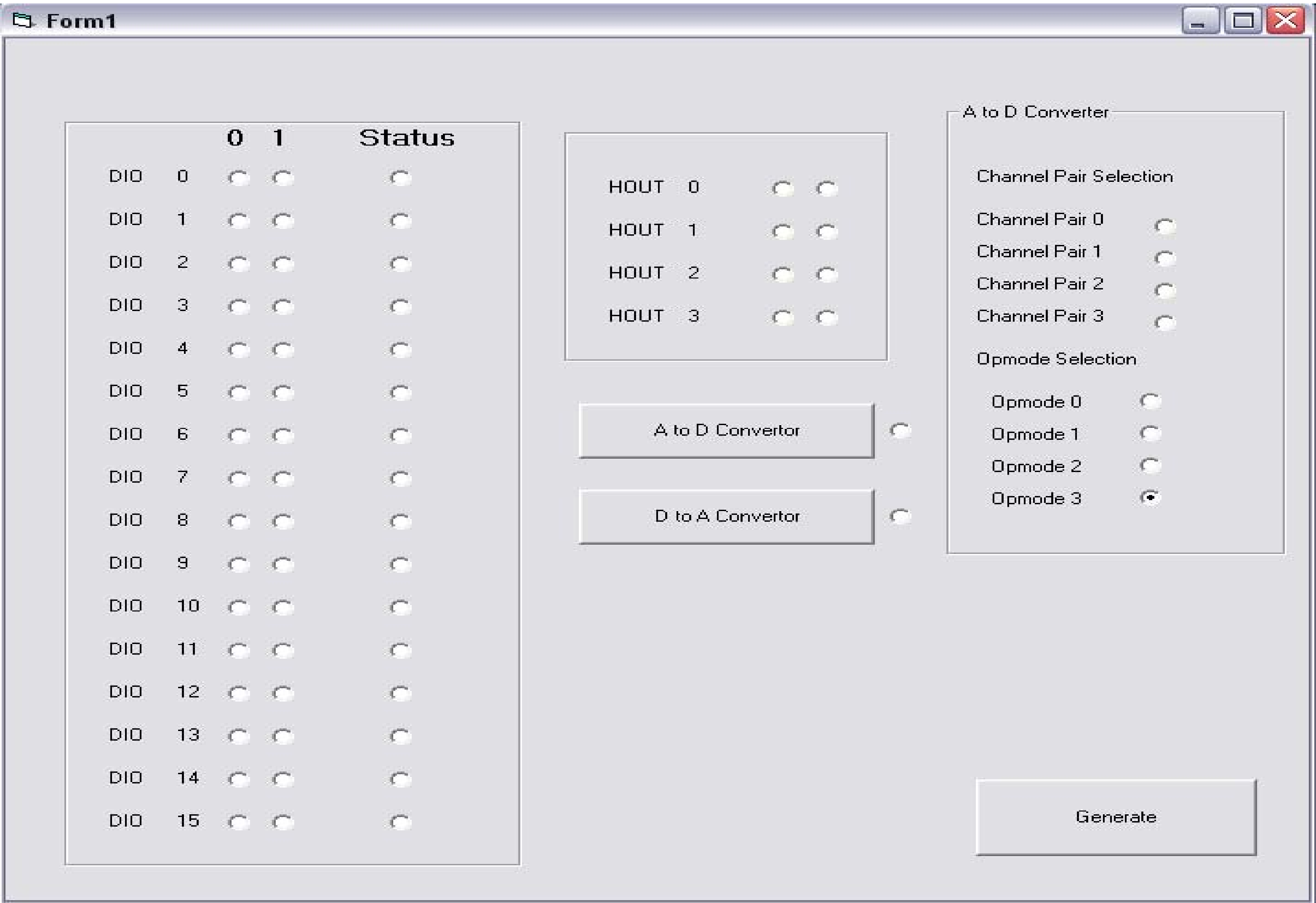
- The four D/A converter outputs are buffered and scaled to provide an output from 0 V to +10 V (12-bit resolution) or ±10 V (11-bit resolution, one bit used for polarity).
- There are also four 4–20 mA current outputs.
- The maximum D/A converter output current is 10 mA per channel for the voltage outputs (in order to minimize power dissipation associated with the D to A converter)

The Graphical User Interface

- The GUI has been designed keeping in mind the ease with which the user can configure the pins on the board .
- The front end of the GUI has been developed using VC++ and the back end part involves generation of a text file comprising of a code in Dynamic C which can be stored with the extension ".C" and then compiled using the Dynamic C compiler.

Contd..

- First column comprises of Digital Input/Output pins referred to as DIO 0-DIO15.
- Third column comprises of 4 High Current outputs that can be configured for either sourcing or sinking and are depicted as HOUT 0- HOUT3.
- There is a separate column for selecting the channel pair and the opmode working condition for the A to D converter.
- There is another column for the selection of the configuration and the mode of operation of the D to A converter.



		0	1	Status
DIO	0	○	○	○
DIO	1	○	○	○
DIO	2	○	○	○
DIO	3	○	○	○
DIO	4	○	○	○
DIO	5	○	○	○
DIO	6	○	○	○
DIO	7	○	○	○
DIO	8	○	○	○
DIO	9	○	○	○
DIO	10	○	○	○
DIO	11	○	○	○
DIO	12	○	○	○
DIO	13	○	○	○
DIO	14	○	○	○
DIO	15	○	○	○

HOUT	0	○	○
HOUT	1	○	○
HOUT	2	○	○
HOUT	3	○	○

A to D Convertor ○

D to A Convertor ○

A to D Converter

Channel Pair Selection

- Channel Pair 0 ○
- Channel Pair 1 ○
- Channel Pair 2 ○
- Channel Pair 3 ○

Opmode Selection

- Opmode 0 ○
- Opmode 1 ○
- Opmode 2 ○
- [✓ Opmode 3] ●

Generate

Backend of the GUI :

- The user will select from the available options on the GUI for the available columns. After selection is done the user will have to press the “Generate” button on the GUI for the actual setting’s to take place.
  If any pin is left unselected, it will be assigned a default value of 0(standard default value for BL2600 subsytems).

- To generate a file containing the code which configures the ports with the selected values or default values, if not selected.

- The file containing the C code will then be executed on the SRAM of the board and hence will set the ports to the desired values.

The Configurable Ports on the BL2600 :

- There are 16 configurable Digital Input/Output pins referred to as DIO0-DIO15.
- There are 4 High Current outputs that can be configured for either sourcing or sinking and are depicted as HOUT0- HOUT3.
- There is a separate column for selecting the channel pair and the opmode working condition for the A to D converter.
- There is another column for the selection of the configuration and the mode of operation of the D to A converter.

The Approach of Dynamic C :

- For BL2600, it specifies a number of in-built functions that should

  be used   for the requirements specified.
- All the in-built functions are defined in the library file BLXX.lib.
- There are separate function calls for each of the functionalities available viz. Digital I/O, Fixed Digital Inputs, A to D and D to A converters and High Current Outputs.

Function’s Used :

1. brdInit() : Board Initialization

   Calling this function at the beginning of program initializes the system I/O ports and loads all the A/D converter and D/A converter calibration constants from flash memory into SRAM for use by the program.

Contd…

2. digOutConfig(int configuration) : Digital Output Configuration
Configures any of the 16 configurable I/O channels to be an output. This configuration information is then used by the digOut function to determine whether a given channel is configured to be an output.
 e.g. configuration = 0x540E
        0x540E = 0101010000001110 (bits 15–0)

3. digOut(int channel, int state) : Digital Output
Sets the state of a configurable I/O channel (DIO00–DIO15) configured as a sinking digital output to a logic 0 or a logic 1. This function only allows control of channels that are configured to be an output by the digOutConfig function.
 Here, channel=0-15 and state= 0 or 1.

Contd…

4. anaInConfig(int ch_pair, int opmode) : Analog Input Configuration
Configures an A/D converter input channel pair for a given mode of operation.
This function must be called before accessing the A/D converter chip.
Parameters:
ch_pair are the channel pairs:
0 = channels 0 and 1
1 = channels 2 and 3
2 = channels 4 and 5
3 = channels 6 and 7
opmode selects the mode of operation for the channel pair on A/D converter:
0 = Single-Ended unipolar (0–10 V)
1 = Single-Ended bipolar (±10 V)
2 = Differential bipolar (±20 V)
3 = 4–20 mA

Contd…

5. anaOutConfig(char configuration, int mode) : Analog Out Configuration
Configures the D/A converter chip for a given output voltage range, 0–10 V or ±10 V, and loads the calibration data for use by the D/A converter API functions. This function must be called before accessing any of the D/A converter channels.

Parameters: configuration sets the output configuration as follows:

0 = unipolar operation. (0–10V and 4–20 mA)

1 = bipolar operation. (±10V and 4–20 mA)

Mode :

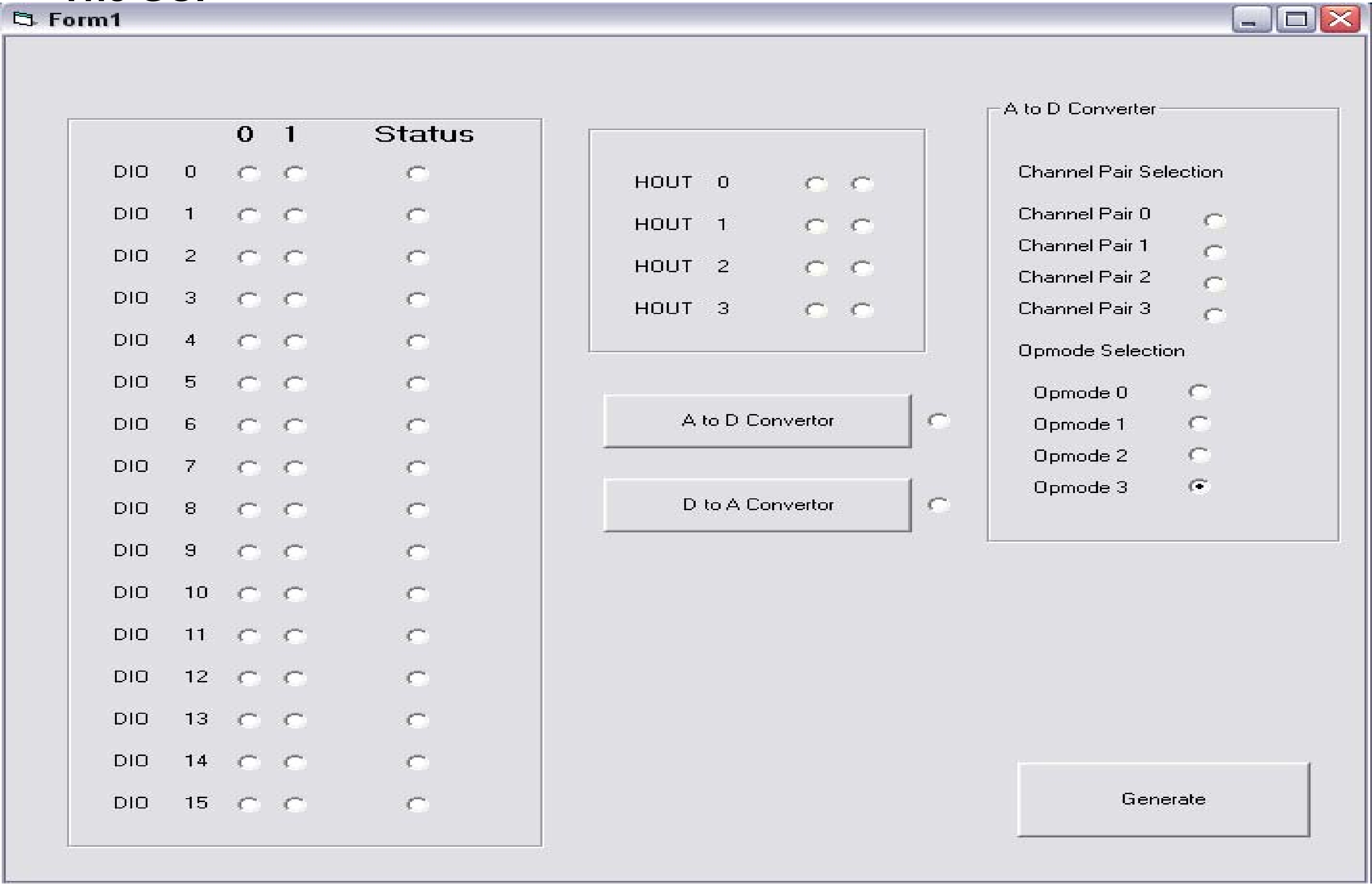
0 = asynchronous—an output is updated at the time data are written to the given channel

1 = synchronous—all outputs are updated with data previously written when the anaOutStrobe function is executed.

Contd…

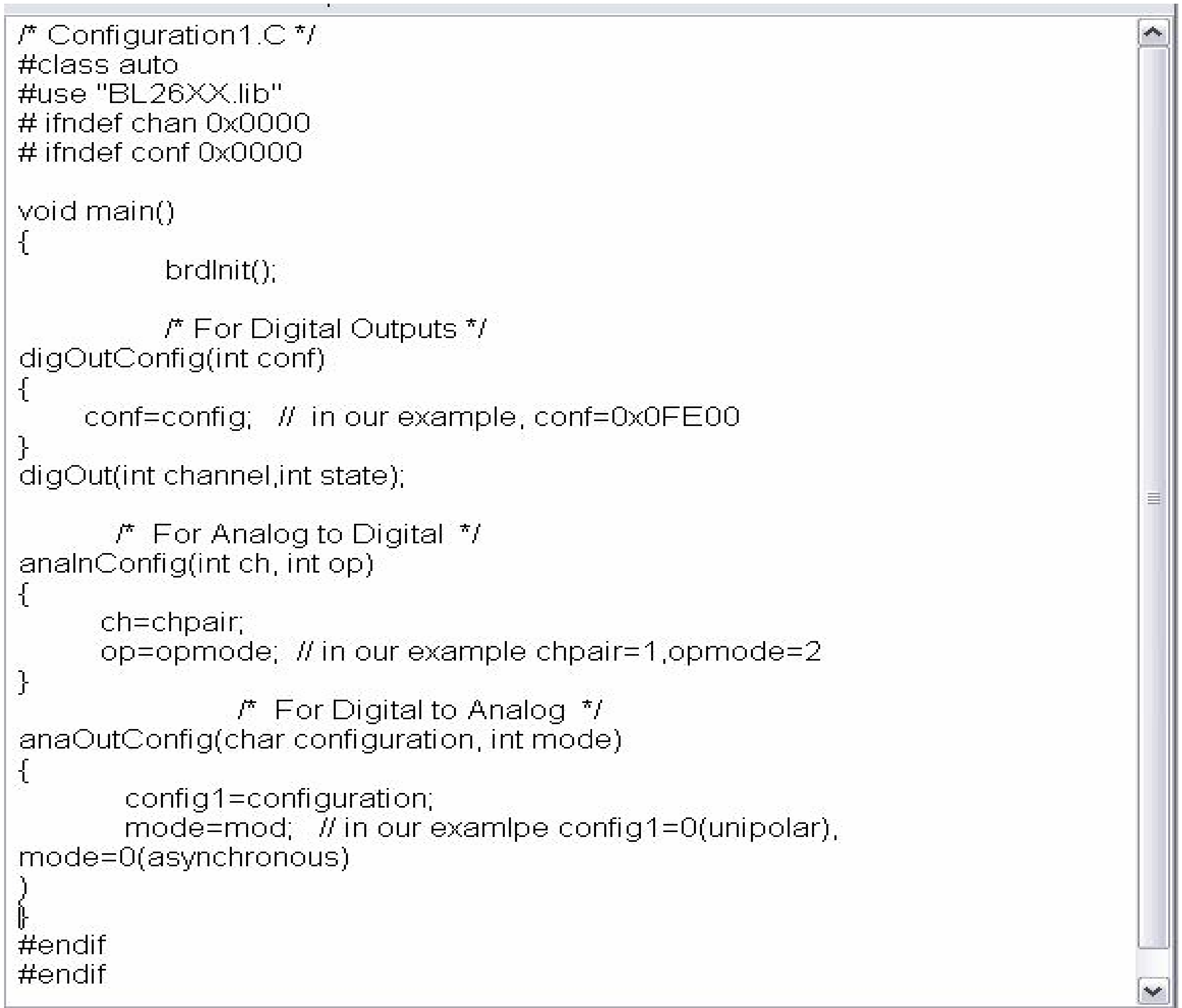
6. digHoutConfig(char configuration) :
Configures a high-current output to be either a sinking or a sourcing output. The configuration options are:
configuration is a 1-byte parameter where 4 bits are used for the high-current outputs HOUT0–HOUT3.
Bit 3 = high-current output channel HOUT3
Bit 2 = high-current output channel HOUT2
Bit 1 = high-current output channel HOUT1
Bit 0 = high-current output channel HOUT0
(bits 4–7 are not used)
The high-current outputs can be configured to be sinking or sourcing outputs by setting the corresponding
bit to an 0 or 1: 0 = sinking, 1 = sourcing.
e.g. configuration=0x0B

The GUI



Opmode Selection: Opmode 0 Opmode 1 Opmode 2 [✓ Opmode 3]

The Generated File



```
/* Configuration1.C */
#class auto
#use "BL26XX.lib"
# ifndef chan 0x0000
# ifndef conf 0x0000

void main()
{
                brdInit();

                /* For Digital Outputs */
digOutConfig(int conf)
{
       conf=config;   //  in our example, conf=0x0FE00
}
digOut(int channel,int state);

           /*  For Analog to Digital  */
analnConfig(int ch, int op)
{
         ch=chpair;
         op=opmode;  // in our example chpair=1,opmode=2
}
                        /*  For Digital to Analog  */
anaOutConfig(char configuration, int mode)
{
           config1=configuration;
           mode=mod;   // in our examlpe config1=0(unipolar),
mode=0(asynchronous)
}
}
#endif
#endif
```

Conclusion

We have successfully created a GUI to configure the pins present on BL2600. The front end of the GUI has been made in VC++ and the backend involves the generation of Dynamic C code as a text file.
In future this can be used to configure the ports and pins without writing the code each time. Therefore one file containing the dynamic C code can be created with the help of the GUI that will be directly executed on the BL2600.