Presentation on

GRAPHIC USER INTERFACE Design for

RCM3400 CONFIGURABILITY

OBJECTIVE

OUR MAIN AIM IS TO CONFIGURE ALL THE PORT PINS OF RCM3400 AS INPUT AND OUTPUT PINS ,USING A DYNAMIC C COMPILER, WHICH CAN BE CONFIGURED EXTERNALLY ALSO WITH THE HELP OF A GUI (using VC++)

CONTENTS

METHODOLOGY

FRONT END PROGRAMMING

We studied VC++ using “VC++ Complete Reference” series of texts and were able to successfully design the layout of the interface.
VC++ has an inbuilt Application Wizard (AppWizard) that was used for this purpose.
We created an application file called “rcm3400-gui” which can be directly operated on any system which contains VC++.

BACK END PROGRAMMING

The back end programming is done in Dynamic C along with linking so that an appropriate text file along with the “to be burnt on board” code is generated.
.

REQUIREMENTS

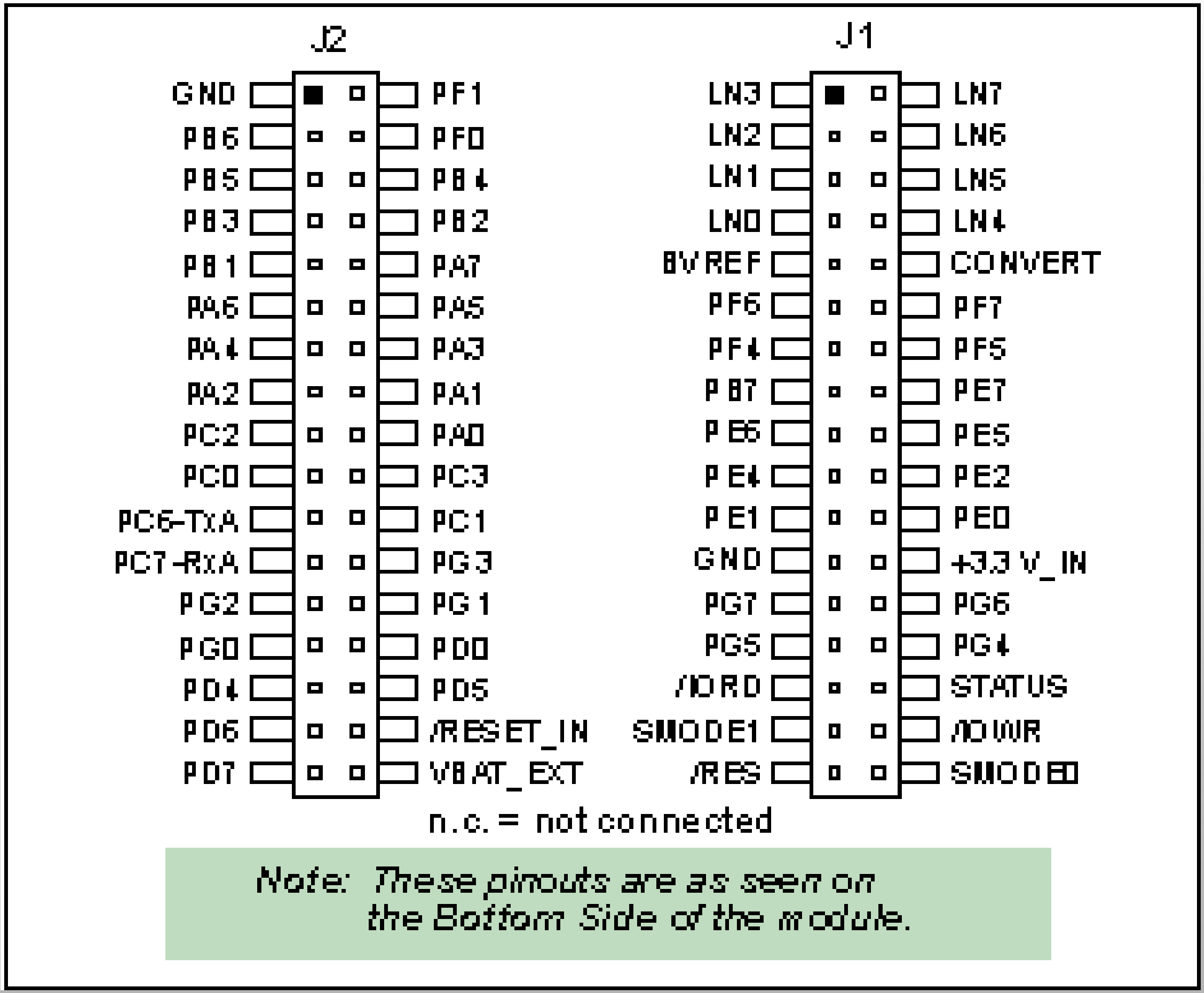
SOFTWARE

- **Dynamic C Compiler**
- **VC++ 6.0 Suite**

HARDWARE

- **RCM3400 Development Kit**

RCM3400 Digital Inputs and Outputs



GRAPHIC USER INTERFACE

DESIGN PARAMETERS

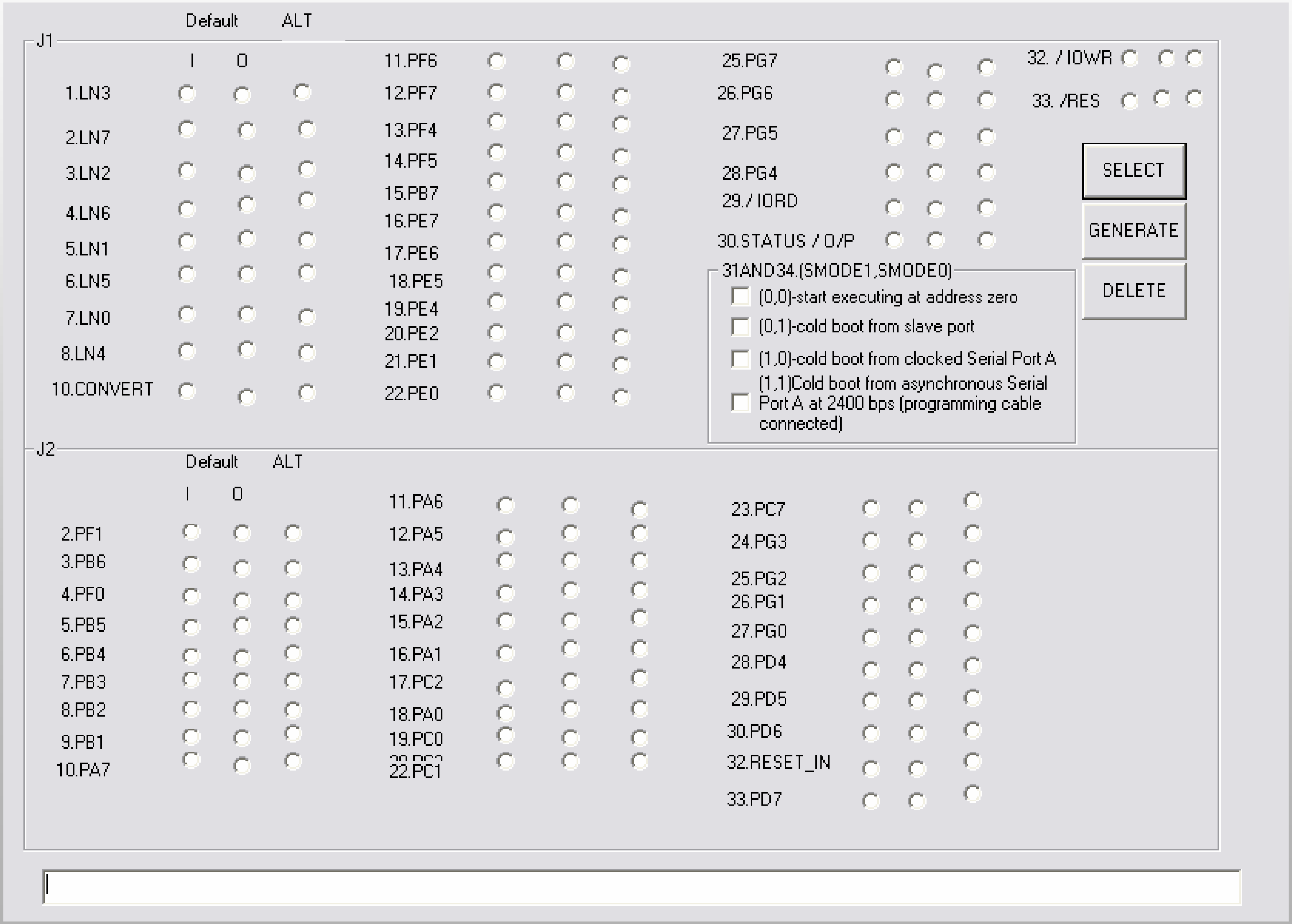
- THE GUI WAS DESIGNED KEEPING IN MIND THE USABILITY OF THE PROGRAMMER

- ALL THE REQUIRED PINS WERE PRESENTED ONTO A SINGLE DESIGN PAGE SO THAT THE PROGRAMMER WOULDN'T HAVE ANY TROUBLES OF SHUNTING BETWEEN PAGES

- THE PINS WERE DISTINCTLY DIVIDED INTO TWO SECTIONS ACCORDING TO THE TWO HEADERS J1 AND J2 AS PRESENT ON THE BOARD FOR EASIER DIFFERENTIATION

GRAPHIC USER INTERFACE

LAYOUT



J1

Default: I, O — ALT

1.LN3
2.LN7
3.LN2
4.LN6
5.LN1
6.LN5
7.LN0
8.LN4
10.CONVERT

11.PF6
12.PF7
13.PF4
14.PF5
15.PB7
16.PE7
17.PE6
18.PE5
19.PE4
20.PE2
21.PE1
22.PE0

25.PG7
26.PG6
27.PG5
28.PG4
29./ IORD
30.STATUS / O/P

32. / IOWR
33. /RES

31AND34.(SMODE1,SMODE0)

- [ ] (0,0)-start executing at address zero
- [ ] (0,1)-cold boot from slave port
- [ ] (1,0)-cold boot from clocked Serial Port A
- [ ] (1,1)Cold boot from asynchronous Serial Port A at 2400 bps (programming cable connected)

J2

Default: I, O — ALT

2.PF1
3.PB6
4.PF0
5.PB5
6.PB4
7.PB3
8.PB2
9.PB1
10.PA7

11.PA6
12.PA5
13.PA4
14.PA3
15.PA2
16.PA1
17.PC2
18.PA0
19.PC0
20.PC3
22.PC1

23.PC7
24.PG3
25.PG2
26.PG1
27.PG0
28.PD4
29.PD5
30.PD6
32.RESET_IN
33.PD7

GRAPHIC USER INTERFACE

HEADER J1

- The initial ln1-ln7 in j1 represent a/d converters of serial port b and can only be configured as inputs

- Pins 11 trough 28 (i.E. The port pins) can be configured both as inputs and outputs and also have alternate uses if one needs to use them

GRAPHIC USER INTERFACE

Memory I/O Interface

- The Rabbit 3000 address lines (A0-A19) and all the data lines (D0-D7) are routed internally to the onboard flash memory and SRAM chips. I/0 write (/IOWR) and I/0 read (/IORD) are available for interfacing to external devices.

- Parallel Port A can also be used as an external I/O data bus to isolate external I/O from the main data bus. Parallel Port B pins PB2-PB7 can also be used as an auxiliary address bus.

- Pins pa’x’ are used as the parallel 8-bit port

GRAPHIC USER INTERFACE

Other Inputs and Outputs

- The status, /RESET_IN, SMODE0, and SMODE1 I/O are normally associated with the programming port.

- /RESET_IN is an external input used to reset the Rabbit 3000 microprocessor and the RCM3400 memory. /RES is an output from the reset circuitry that can be used to reset other peripheral devices.

- The two SMODE pins, SMODE0 and SMODE1, are available as inputs. The logic state of these two pins determines the startup procedure after a reset.

GRAPHIC USER INTERFACE

Serial Ports

- There are five serial ports designated as Serial Ports A, C, D, E, and F. All five serial ports can operate in an asynchronous mode up to the baud rate of the system clock divided by 8.
- Serial Port A is normally used as a programming port.
- Serial Port B is used by the A/D converter.
- Serial Ports C and D can also be operated in the clocked serial mode.
- Serial Ports E and F can also be configured as SDLC/HDLC serial ports.

GRAPHIC USER INTERFACE

Back End programming Functions

- Board Initialization - void brdInit (void);
- Digital I/O - WrPortI(P'X'DDR, &P'X'DDRShadow, 0x00);
- AD converters configuration – unsigned int anaInConfig(unsigned int instructionbyte, unsigned int cmd, long baud)

GRAPHIC USER INTERFACE

HEADER J2

- ALMOST ALL PINS IN THIS HEADER HAVE INPUT/OUTPUT AND ALTERNATE USES
- PINS PC’X’ CAN ONLY BE USED FOR INPUT OR OUTPUT AS GIVEN BY THE MANUAL
- PINS PA’X’ ARE USED AS THE PARALLEL 8-BIT PORT

GRAPHIC USER INTERFACE

An application called “rcm3400-gui” was created.

rcm3400-gui.dsp

This file (the project file) contains information at the project level and is used to build a single project or subproject. Other users can share the project (.dsp) file, but they should export the make files locally.

rcm3400-gui.h

This is the main header file for the application. It includes other project specific headers (including Resource.h) and declares the Crcm3400-guiApp application class.

GRAPHIC USER INTERFACE

DESIGN PROCEDURE AND MODULES

rcm3400-gui.cpp

This is the main application source file that contains the application class C rcm3400-guiApp.

rcm3400-gui.rc

This is a listing of all of the Microsoft Windows resources that the program uses. It includes the icons, bitmaps, and cursors that are stored in the RES subdirectory. This file can be directly edited in Microsoft Visual C++.

rcm3400-gui.clw

This file contains information used by ClassWizard to edit existing classes or add new classes. ClassWizard also uses this file to store information needed to create and edit message maps and dialog data maps and to create prototype member functions.

GRAPHIC USER INTERFACE

DESIGN PROCEDURE AND MODULES

res\ rcm3400-gui.ico

This is an icon file, which is used as the application's icon. This icon is included by the main resource file rcm3400-gui.rc.

res\ rcm3400-gui.rc2

This file contains resources that are not edited by Microsoft Visual C++. You should place all resources not editable by the resource editor in this file.

MainFrm.h, MainFrm.cpp

These files contain the frame class CMainFrame, which is derived from CFrameWnd and controls all SDI frame features.

GRAPHIC USER INTERFACE

res\Toolbar.bmp

This bitmap file is used to create tiled images for the toolbar. The initial toolbar and status bar are constructed in the CMainFrame class. Edit this toolbar bitmap using the resource editor, and update the IDR_MAINFRAME TOOLBAR array in rcm3400-gui.rc to add toolbar buttons.

rcm3400-guiDoc.h, rcm3400-guiDoc.cpp - the document

These files contain your C rcm3400-guiDoc class. Edit these files to add your special document data and to implement file saving and loading (via C rcm3400-guiDoc::Serialize).

rcm3400-guiView.h, rcm3400-guiView.cpp - the view of the document These files contain your Crcm3400-guiView class. Crcm3400-guiView objects are used to view Crcm3400-guiDoc objects.

GRAPHIC USER INTERFACE

DESIGN PROCEDURE AND MODULES

Other standard files:

StdAfx.h, StdAfx.cpp

These files are used to build a precompiled header (PCH) file named rcm3400-gui.pch and precompiled types file named StdAfx.obj.

Resource.h

This is the standard header file, which defines new resource IDs. Microsoft Visual C++ reads and updates this file.

RESULTS

We were successfully able to fabricate a “Graphics User Interface” which allows us to easily configure the pins based on the RCM3400 single board computer. This GUI will intern make the task of configuring the board according to one’s needs a whole lot easier. Here we used the VC++ platform to design a user friendly interface where you can assign the configurable pins the specific task that you want them to perform. Next a text file is generated which can later be burnt onto the board for permanent/temporary settings.

CONCLUSION and SCOPE

The GUI was completed successfully along with the text export options which exports it into a text box to be later burnt onto the board.

Many other enhancements can be made into the code such as :-

- Wi-Fi operations
- Bluetooth Support
- Serial/Parallel Transfer

THANK YOU