Accessing and Writing to the TMS320C2xx I/O Memory Mapped Registers

Literature Number: BPRA062
Texas Instruments Europe
July 1997

Contents

List of Figures

Accessing and Writing to the TMS320C2xx I/O Memory Mapped Registers

ABSTRACT

The TMS320C2xx generation from Texas Instruments brings the power of digital signal processing to designers of high performance, cost sensitive, emerging applications. Combining high performance with ultra-low cost, the ’C2xx generation offers the most optimal balance of price and performance of any DSP in the industry.

1. Introduction

The TMS320C2xx is a family of 16 bit fixed-point CMOS DSPs. Each generation varies slightly in peripheral configuration as can be seen from the following figures and descriptions.

1.1 The TMS320C203

256 words of single-access data/program RAM and 288 words of dual-access data RAM.

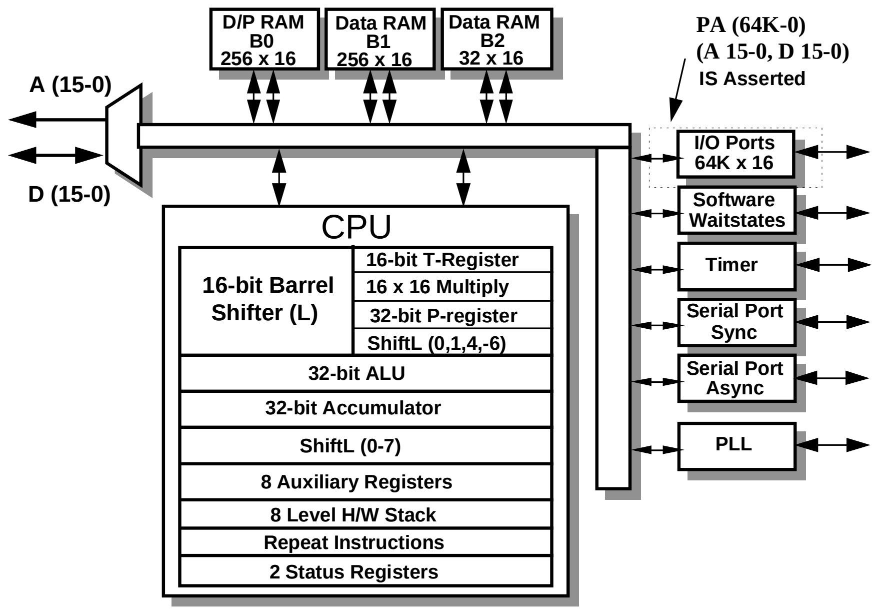


Figure 1: Layout of the TMS320C203

1.2 The TMS320C204

256 words of single-access data/program RAM and 288 words of dual-access data RAM and 4K words of ROM.

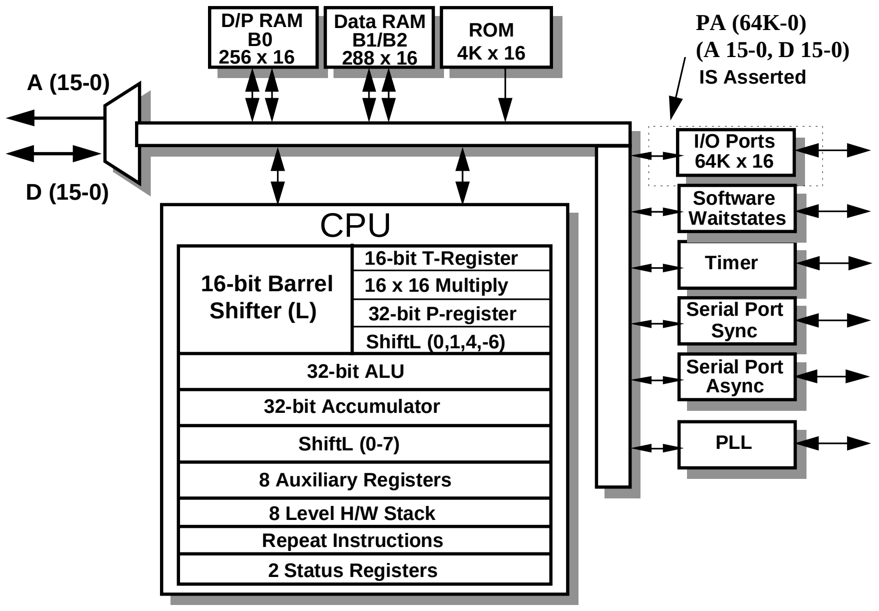


Figure 2: Layout of the TMS320C204

1.3 The TMS320C205

4K words of single-access data/program RAM, 256 words of dual-access data/program RAM and 288 words of dual-access data RAM

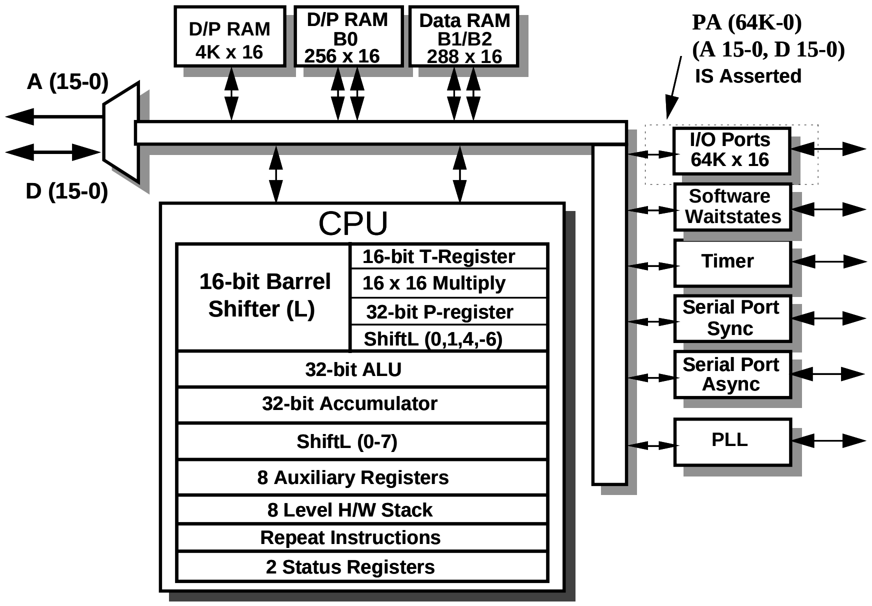


Figure 3: Layout of the TMS320C205

1.4 The TMS320F206

4K words of single-access data/program RAM, 256 words of dual-access data/program RAM and 288 words of dual-access data RAM and 32K words of ROM.

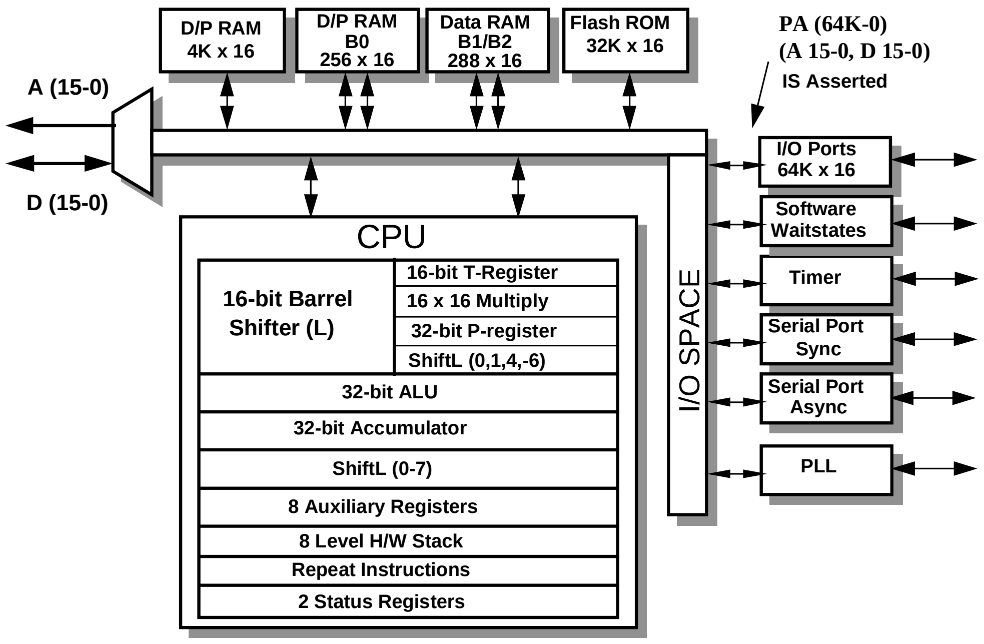


Figure 4: Layout of the TMS320F206

1.5 The TMS320F207

4K words of single-access data/program RAM, 256 words of dual-access data/program RAM and 288 words of dual-access data RAM and 32K words of ROM.

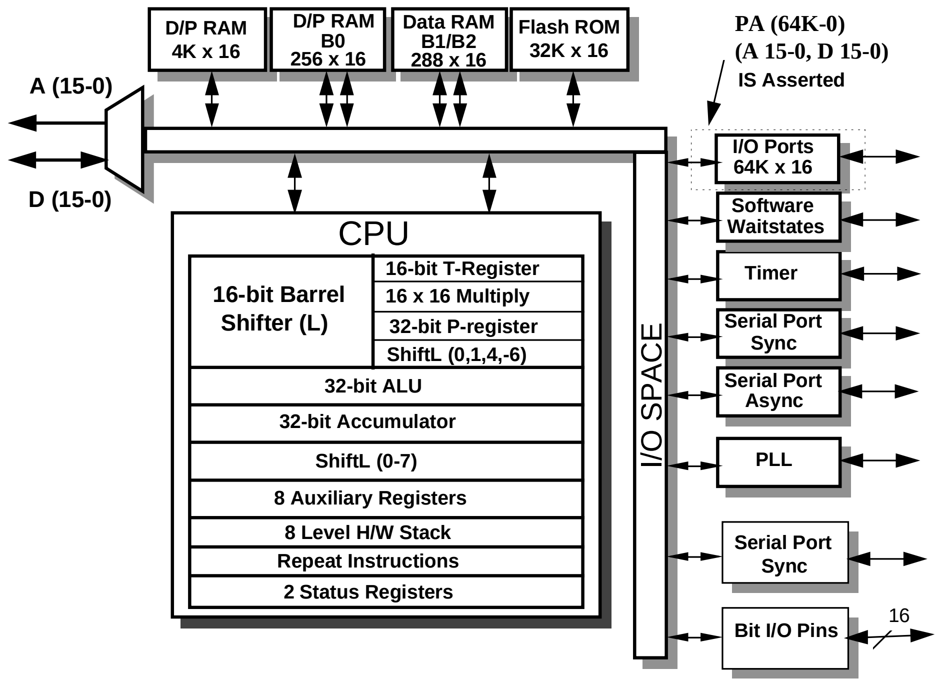


Figure 5: Layout of the TMS320F207

1.6 The TMS320C209

4K words of single-access data/program RAM, 256 words of dual-access data/program RAM, 288 words of dual-access data RAM and 4K words of ROM.

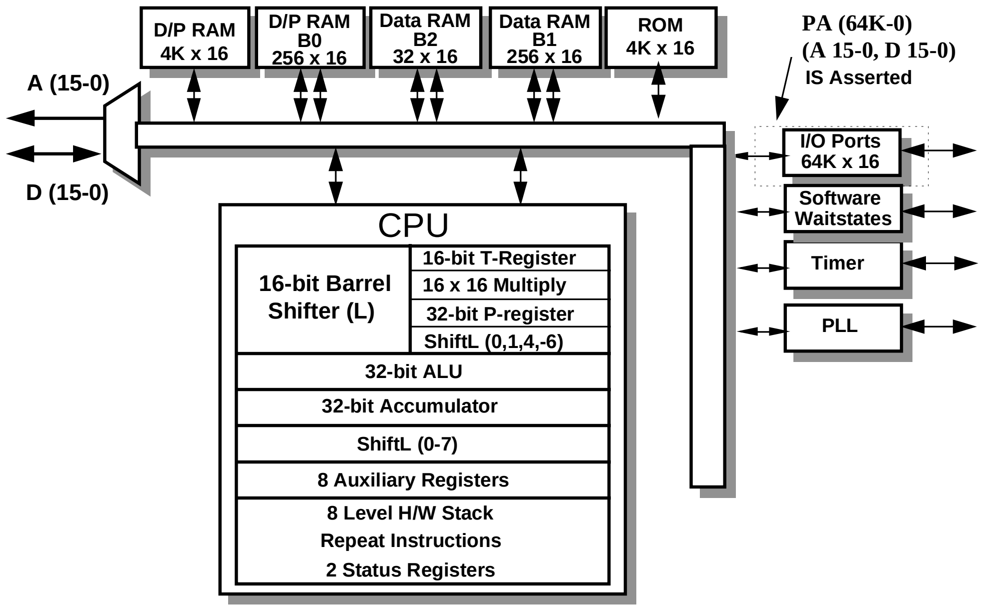


Figure 6: Layout of the TMS320C209

The idea of this application report is to provide the user with simple tools and explanations on how to easily access the registers mapped to I/O memory of the TMS320C203 and the TMS320C209 DSP using C.

2. The I/O Memory Mapped Registers of the TMS320x2xx

This chapter will discuss the memory location and layout of the various members of the TMS320x2xx family I/O memory-mapped control and data registers. By setting and clearing bits on these registers, the on-chip peripherals can be easily reconfigured.

2.1 I/O Memory-Mapped Registers For Controlling On-Chip Peripherals

Most of the registers that control the on-chip peripherals are mapped to internal I/O memory space. These registers are used to set up the various parameters required when communicating with the outside world. All the TMS320x2xx family have the same register addresses apart from the ‘C209. Figure 7 describes the location in memory, the name and the function of each of the Memory-Mapped Registers in the TMS320x2xx family.

Addresses on ‘x2xx	Address on ‘C209 only	Name	Description
FFE8h	-	CLK	Clock out register. Enable/disable CLKOUT1 pin
FFECh	-	IC	Interrupt Control register
FFF0h	-	SDTR	Synchronous serial port transmit and receive register
FFF1h	-	SSPCR	Synchronous serial port register
FFF4h	-	ADTR	Asynchronous serial port transmit and receive register
FFF5h	-	ASPCR	Asynchronous serial port register
FFF6h	-	IOSR	Input/output status register
FFF7h	-	BRD	Baud rate divisor register
FFF8h	FFFCh	TCR	Timer control register
FFF9h	FFFDh	PRD	Timer period register
FFFAh	FFFEh	TIM	Timer counter register
FFFCh	FFFFh	WSGR	Wait-state generator control register

Figure 7: I/O Memory Location of the ‘x2xx Family Control Registers

In order to set the various bits in the registers, the layout of the registers is required. The next part of this section defines the structures of the I/O memory-mapped registers for the TMS320x2xx, excluding the ‘C209 DSP.

2.2 I/O Memory-Mapped Registers Layout for all TMS320x2xx except ‘C209

If further explanations on these register functions are required, please refer to the TMS320C2xx User’s Guide.

2.2.1 CLK Register

15-1	0
Reserved	CLKOUT1
0	R/W

Figure 8: CLK Register Layout

2.2.2 IC Register

15-5	4	3	2	1	0
Reserved	MODE	FINT3	FINT2	MINT3	MINT2
0	R/W	R/W	R/W	R/W	R/W

Figure 9: Interrupt Control Register Layout

2.2.3 SDTR Register

The SDTR is the Synchronous Serial Port Transmit and Receive Register

2.2.4 SSPCR Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FREE	SOFT	TCOMP	RFNE	FT1	FT0	FR1	FR0	IN1	IN0	XRST	RRST	TXM	MCM	FSM	DLB
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Figure 10: Synchronous Serial Port Register Layout

2.2.5 ADTR Register

The ADTR is the Asynchronous Serial Port Transmit and Receive Register

2.2.6 ASPCR Register

15	14	13	12-10	9	8	7	6	5	4	3	2	1	0
FREE	SOFT	URST	Reserved	DIM	TIM	RIM	STB	CAD	SETBRK	CIO3	CIO2	CIO1	CIO0
R/W	R/W	R/W	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 11: Asynchronous Serial Port Register Layout

2.2.7 IOSR Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Resvd	ADC	BI	TEMT	THRE	FE	OE	DR	DIO3	DIO2	DIO1	DIO0	IO3	IO2	IO1	IO0
0	R/W	R/W	R	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R	R	R	R

Figure 12: Input/output Status Register Layout

2.2.8 BRD Register

The Baud Rate Divisor Register determines the baud rate of the asynchronous serial port.

2.2.9 TCR Register

15-12	11	10	9-6	5	4	3-0
Reserved	FREE	SOFT	PSC	TRB	TSS	TDDR
0	R/W	R/W	R/W	R/W	W	W

Figure 13: Timer Control Register Layout

2.2.10 PRD Register

The Timer Period Register is 16 bits wide. It holds the next starting count for the timer and supplies TIM with the next value to decrement.

2.2.11 TIM Register

The Timer Counter Register is 16 bits wide and holds the current count of the timer.

2.2.12 WSGR Register

15-12	11-9	8-6	5-3	2-0
Reserved	ISWS	DSWS	PSUWS	PSLWS
0	W	W	W	W

Figure 14: Wait-State Generator Control Register Layout

2.3 I/O Memory-Mapped Registers Layout for the TMS320C209

The TMS320C209 has a different register memory-map than the rest of the TMS320x2xx devices. The four register that are I/O memory-mapped are as follows :

2.3.1 TCR Register

15-10	9-6	5	4	3-0
Reserved	PSC	TRB	TSS	TDDR
0	R/W	R/W	W	W

Figure 15: Timer Control Register Layout

2.3.2 PRD Register

The Timer Period Register is 16 bits wide. It holds the next starting count for the timer and supplies TIM with the next value to decrement.

2.3.3 TIM Register

The Timer Counter Register is 16 bits wide and holds the current count of the timer.

2.3.4 WSGR Register

15-4	3	2	1	0
Reserved	AVIS	ISWS	DSWS	PSWS
0	W	W	W	W

Figure 16: Wait-State Generator Control Register Layout

3. Accessing I/O Memory-Mapped Registers

In order to access the various registers, modules have been written in C and Assembly Language to facilitate their initialization and modification. The next chapter will deal with the way in which data is passed to and from the corresponding functions. The relevant sections of code will be listed and explained along with a description of how to use the functions.

3.1 The C Functions

There are two main functions in C that deal with the writing of data to the registers and they are compiled into the run-time libraries x2xxio.lib or the c209io.lib :

- Port_Value = “Register name”_reg_value(, , , , , , ...)
- port_function(“Register name”,Port_Value)

The first function simply returns a value calculated from the list of parameters included in the brackets and the second writes the calculated value to the desired I/O-Memory-Mapped Register.

3.1.1 Calculating Required Register Values

As shown by the figures in the previous chapter, each register is divided into a number of distinct sections. The desired values are passed to the function in exactly the same way that the register is set-up. The list of the 7 functions created for this purpose is as follows:

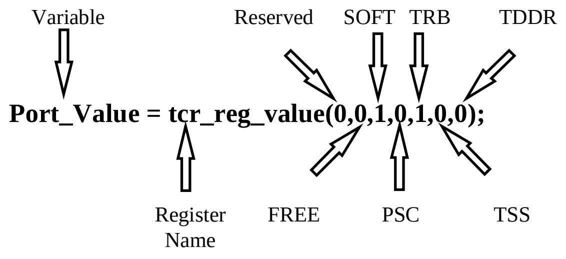
1. clk_reg_value(Reserved,CLKOUT1)
2. ic_reg_value(Reserved,MODE,F3,F2,M3,M2)
3. sspcr_reg_value(FREE,SOFT,TCOMP,RFNE,FT1,FT0,FR1,FR0,IN1,IN0,XRST,RRST,TXM,MCM,FSM,DLB)
4. aspcr_reg_value(FREE,SOFT,URST,Reserved,DIM,TIM,RIM,STB,CAD,SETBRK,CIO3,CIO2,CIO1,CIO0)
5. iosr_reg_value(Reserved,ADC,BI,TEMT,THRE,FE,OE,DR,DIO3,DIO2,DIO1,DIO0,IO3,IO2,IO1,IO0)
6. tcr_reg_value(Reserved,FREE,SOFT,PSC,TRB,TSS,TDDR)
7. wsgr_reg_value(Reserved,ISWS,DSWS,PSUWS,PSLWS)

A typical example for using these functions would be as follows:

E.g. Using the TCR Register:

15-12	11	10	9-6	5	4	3-0
Reserved	FREE	SOFT	PSC	TRB	TSS	TDDR
0	R/W	R/W	R/W	R/W	W	W

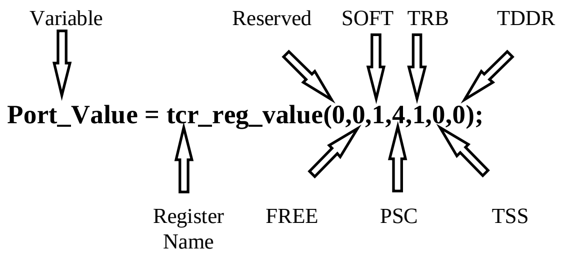
If only SOFT and TRB are to be set to 1 then the function will look like:



The function would then return the value 1056 or 420h which is stored in the global variable Port_Value.

One important thing to remember when passing values to these functions is that the value should be in decimal form.

i.e. if a section of a register has a bit field length greater than one (e.g.PSC section of the TCR register), then the value passed takes the decimal form.



This would be equivalent to PSC being set to “0100”in binary, which is 4 in decimal.

As for the remaining I/O Memory-Mapped Registers:

1. SDTR
2. ADTR
3. BRD
4. PRD
5. TIM

Because these register are simply 16 bit wide, they do not require this function to return a value. They can be programmed directly using the function described below.

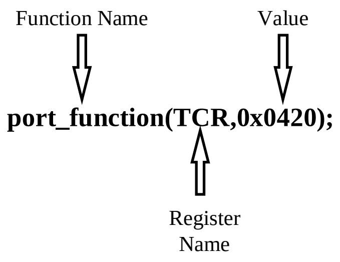
3.1.2 *Writing Values to I/O Memory-Mapped Registers*

In performing this action, only the name of the register and the desired value are passed to the function. This can be done in two ways.

1. Passing the register name and a fixed value.
2. Passing the register name and the calculated Port_Value.

3.1.2.1 Passing the Register Name and a Fixed Value

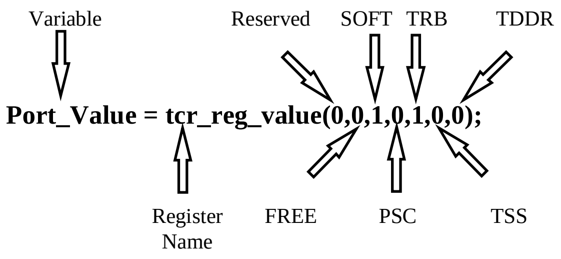
Setting the TCR Register to 0420h:



```
port_function(TCR,0x0420);
```

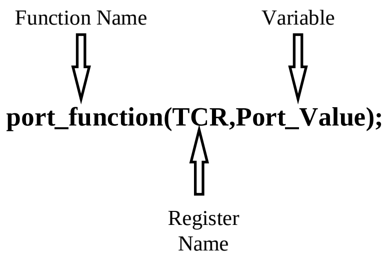
3.1.2.2 Passing the Register Name and the calculated Port_Value

Setting the TCR register to a value calculated by the function:



```
Port_Value = tcr_reg_value(0,0,1,0,1,0,0);
```

And finally, to write to the register the following must be used:



```
port_function(TCR,Port_Value);
```

The port_function can be used to write data to all the I/O Memory-Mapped Registers in the TMS320x2xx family as listed below:

1. CLK
2. IC
3. SDTR
4. SSPCR
5. ADTR
6. ASPCR
7. IOSR
8. BRD

9. TCR
10. PRD
11. TIM
12. WSGR

The complete source code for the C functions can be found in Appendix A.

3.2 The Assembly Function

It is not actually necessary to know how the ASM functions. However, for the purpose of this report it may be of interest for the user to develop further ASM functions interfaced to C function calls.

3.2.1 The Simple Assembly Function Call

When interfacing ASM functions to C function calls, the following example shows the simplest interface procedure:

3.2.1.1 The C Procedure Function Call

```
main
{
        out_to_port ();
}
```

3.2.1.2 The ASM Function

```
; *******************************
; * Function called from C code *
; *******************************
        .text                           ; Section in defined in Data memory
_out_to_port:                           ; Function called from C
        POPD    *+                      ; Place return address on stack
        SAR     AR0,*+                  ; Store AR0 Frame Pointer
        SAR     AR1,*                   ; Store AR1 Stack Pointer

        ; Place code here using AR2 to access data if necessary

        ; User code stops here

        MAR     *,AR1                   ; Select AR1
        SBRK    #1                      ;
        LAR     AR0,*-                  ; Load AR0 from Stack
        PSHD    *                       ; Load Return Address from Stack
        RET                             ; Return from function call

; *******************************
```

When calling assembly functions from C, it is important to use the software stack to save:

1. The return address
2. The Frame Pointer
3. The Stack Pointer
4. Various other passed variables etc...

Once this has been done, the user can insert the code. On returning to C, the Frame Pointer and the Return Address have to be retrieved from the software stack. Then the function can return to the C calling procedure.

For further information concerning ASM and C interfacing, please refer to the Optimizing C Compiler User's Guide.

The complete source code for the Assembly functions can be found in Appendix B.

3.3 3.3 The Run-Time Library

The x2xxio.lib or the c209io.lib library file must be included in the linker batch file when the C code is compiled. This library contains the ASM function to write to the I/O Memory-Mapped registers as well as the C functions that return the calculated Port_Value.

The Run-Time Library files are :

1. x2xxio.src (Source)
2. x2xxio.lib (Library)
3. c209io.src (Source)
4. c209io.lib (Library)

The complete source code for the Library functions can be found in Appendix C.

3.3.1 Creating and Archive File

In order to create an archive file (i.e. a *.src file, the DSPAR.EXE program must be used). This is the first step to making a library file, which will be compiled with the usercode. When creating such a file, the location of the DSPAR.EXE program is stated in the path. Once this is done, the *.scr file is made by typing the following text at the DOS prompt:

dspar a *archive_name*.src *xxx*.* *xxxx*.* .... (filenames to be added, e.g. *xxxx*.h *xxxx*.c ...)

3.3.2 Creating a Library

Now, to create the compilable library file, the DSPMK.EXE program must be used. Again, the location of the DSPMK.EXE program should is stated in the path and the *.lib file is made by typing the following text at the DOS prompt:

dspmk -v2xx *archive_name*.src

One little note, when creating the *.lib file, the files included in the *.src file must be removed from that directory, otherwise an error will occur. The best thing to do it to move the files listed within the *.src file to a different location and then type:

dspmk -v2xx *archive_name*.src

3.4 Compiling the Code

Now that the library file has been created, it can be included when all the source code is compiled. It contains all the C and ASM functions related to the application note.

This can be done in the same way as for the batch file, as described in Appendix D.2. The DSPCL.EXE program compiles the code, whilst the -v2xx option, reminds the compiler that the code should be compiled for the TMS320C2xx family. The code is then linked using the userc2xx.cmd command file as shown is Appendix D.1.

Note that the library file *.lib has been included at this stage.

4. The Simulator

This code was developed using the TMS320C2x/C2xx/C5x Optimizing C Compiler User's Guide, the TMS320C1x/C2x/C2xx/C5x Assembly Language Tools User's Guide and the 'C2xx Simulator Version 1.30.

In order to use the simulator, the command file memory setup for the linker and the simulator had to be consistent. Additionally, the simulator command file was configured to simulate the necessary registers in I/O memory.

The complete software listing for the:

1. Linker Command file
2. Simulator Command file
3. Simulator initialisation log file

can be found in Appendix D.

5. Implementation

To use the software the following files must be available:

1. User's main C Code	(usercode.c)
2. Header File	(c203io.h / c209io.h)
3. Library File	(c203io.lib / c209io.lib)
4. Command File	(userc2xx.cmd)

For the simulator:

1. Command File	(simuser.cmd)
2. Log File	(userinit.log)

Once the user has written the *usercode* in C, the code must be compiled and linked making sure that the header file c203io.h or c209io.h is in the same directory. Within the linker, the library file c203io.lib or c209io.lib must be included as well. Once the complete user.out file has been generated, it can be loaded into the simulator.

The complete source code for the above files can be seen in Appendices A, C and D.

5.1 Further Development

For the moment, since the C compiler only differentiates between the ’C2x/C2xx/C5x and not yet between the different generations within in family, it is necessary use the corresponding ’C2xx run-time library. Once the compiler has this capability, just making one ’C2xx run-time library will be a simple formality with #ifdef #endif statements. However, until that time has arrived, the user must include the correct ’C2xx family generation file *.h / *.lib when compiling.

References

1. TMS320C2x/C2xx/C5x Optimizing C Compiler	User’s Guide 1995
2. TMS320C2xx	User’s Guide 1997
3. TMS320C1x/C2x/C2xx/5x Assembly Language Tools	User’s Guide 1995
4. TMS320C5x C Source Debugger	User’s Guide 1994
5. TMS320C5x DSP Starter Kit	User’s Guide 1996

Appendix A - C Source Code Listing

A.1 Usercode Code : usercode.c

```
/************************************************
* Example C Program for writing values to      *
* I/O Memory-Mapped Registers for x2xx         *
*                                              *
* Written by Nicholas Holland                  *
*            Texas Instruments                 *
*            1996                              *
*                                              *
* Release Version 1.0                          *
*                                              *
************************************************/

#include "x2xxio.h"                /* Header file with offsets for x2xx */
                                   /* Change for C209 to C209IO.H ...   */

int     Port_Offset;               /* Global variable */
int     Port_Value;                /* Global variable */

/* Program */
void main()
{
        /* The following code shows the normal procedure to define a value
           to a IO register and then to send that value to the register */

        Port_Value = ic_reg_value(0,1,1,1,1,0);
        port_function(IC,Port_Value);           /* function(Reg Name,Reg Value) */

       Port_Value = tcr_reg_value(0,0,0,0,1,1,1);
        port_function(TCR,Port_Value);          /* function(Reg Name,Reg Value) */

       Port_Value = wsgr_reg_value(0,4,1,1,1);
        port_function(WSGR,Port_Value);         /* function(Reg Name,Reg Value) */

       Port_Value = aspcr_reg_value(0,1,0,0,1,1,1,1,1,1,1,1,1,1);
        port_function(ASPCR,Port_Value);        /* function(Reg Name,Reg Value) */
}
```

A.2 Functions Code ‘x2xx : x2xxio.c

```
/************************************************
* Example C functions for IO registers for x2xx *
*                                              *
* Written by Nicholas Holland                  *
*            Texas Instruments                 *
*            1996                              *
*                                              *
* Release Version 1.0                          *
*                                              *
************************************************/

/* The first function listed below is for sending
   a value to a register. The function then calls
   another function in assembly language */

extern int Port_Offset;
extern int Port_Value;

void port_function(int Reg_Offset, int Reg_Value)
{
       Port_Offset = Reg_Offset;       /* Sets Port Offset value */
       Port_Value = Reg_Value;         /* Sets Port value */

       out_to_port();                  /* Calls the ASM function */
}

/************************************************/

/* SDTR ADDR are transmit registers that take a 16 bit word */
/* PRD TIM are counter registers that take a 16 bit word    */
/* BRD determines the baud rate and takes a 16 bit word     */
/* The functions below return complete 16 bit values for
   the corresponding registers                          */

/************************************************/

#ifdef _TMS320C2XX

int clk_reg_value(int Reserved, int CLKOUT1)
{
       int x = 0;

       x = ((Reserved<<1)|(CLKOUT1));
       return (x);

}
/************************************************/
int ic_reg_value(int Reserved,int MODE,int FINT3,int FINT2,int MINT3,
                      int MINT2)
{
       int x = 0;

       x = ((Reserved<<5)|(MODE<<4)|(FINT3<<3)|
                      (FINT2<<2)|(MINT3<<1)|(MINT2));
       return (x);
}
/************************************************/
int sspcr_reg_value(int FREE,int SOFT,int TCOMP,int RFNE,int FT1,int FT0,
                      int FR1,int FR0,int IN1,int IN0,int XRST,int RRST,
                       int TXM,int MCM,int FSM,int DLB)

{
       int x = 0;
       x = ((FREE<<15)|(SOFT<<14)|(TCOMP<<13)|(RFNE<<12)|(FT1<<11)|
                      (FT0<<10)|(FR1<<9)|(FR0<<8)|(IN1<<7)|(IN0<<6)|
                       (XRST<<5)|(RRST<<4)|(TXM<<3)|(MCM<<2)|(FSM<<1)|(DLB));
       return (x);
}
/************************************************/
int aspcr_reg_value(int FREE,int SOFT,int URST,int Reserved,int DIM,int TIM,
                      int RIM,int STB,int CAD,int SETBRK,int CIO3,int CIO2,
                      int CIO1,int CIO0)
{
       int x = 0;
       x = ((FREE<<15)|(SOFT<<14)|(URST<<13)|(Reserved<<10)|(DIM<<9)|
                      (TIM<<8)|(RIM<<7)|(STB<<6)|(CAD<<5)|(SETBRK<<4)|
                      (CIO3<<3)|(CIO2<<2)|(CIO1<<1)|(CIO0));
       return (x);
}
/************************************************/
int iosr_reg_value(int Reserved,int ADC,int BI,int TEMT,int THRE,int FE,
                      int OE,int DR,int DIO3,int DIO2,int DIO1,int DIO0,
                      int IO3,int IO2,int IO1,int IO0)
{
       int x = 0;
       x = ((Reserved<<15)|(ADC<<14)|(BI<<13)|(TEMT<<12)|(THRE<<11)|
                      (FE<<10)|(OE<<9)|(DR<<8)|(DIO3<<7)|(DIO2<<6)|
                      (DIO1<<5)|(DIO0<<4)|(IO3<<3)|(IO2<<2)|(IO1<<1)|(IO0));
       return (x);
}
/************************************************/
int tcr_reg_value(int Reserved,int FREE,int SOFT,int PSC,
                      int TRB,int TSS,int TDDR)
{
       int x = 0;
        x = ((Reserved<<12)|(FREE<<11)|(SOFT<<10)|(PSC<<6)|(TRB<<5)|
                       (TSS<<4)|(TDDR));
       return (x);
}
/************************************************/
int wsgr_reg_value(int Reserved,int ISWS,int DSWS,int PSUWS,int PSLWS)
{
       int x = 0;
       x = ((Reserved<<12)|(ISWS<<9)|(DSWS<<6)|
                      (PSUWS<<3)|(PSLWS));
       return (x);
}
```

```
#endif
```

A.3 Header File ‘x2xx : x2xxio.h

```
/************************************************
* TMS320x2xx Processor Header File x2xx         *
*           This file contains the necessary    *
*           Offsets for the out_port function   *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
*************************************************/

/* Prototypes found in usercode.c */
extern  void out_to_port(void);    /* External asm function */
void port_function(int Reg_Offset, int Reg_Value);

/* Functions to set registers */
/* Functions can be found in x2xxio.lib */
int clk_reg_value(int Reserved,int CLKOUT1);
int ic_reg_value(int Reserved,int MODE,int FINT3,int FINT2,
                      int MINT3,int MINT2);
int sspcr_reg_value(int FREE,int SOFT,int TCOMP,int RFNE,int FT1,int FT0,
                      int FR1,int FR0,int IN1,int IN0,int XRST,int RRST,
                      int TXM,int MCM,int FSM,int DLD);
int aspcr_reg_value(int FREE,int SOFT,int URST,int Reserved,int DIM,int TIM,
                      int RIM,int STB,int CAD,int SETBRK,int CIO3,int CIO2,
                      int CIO1,int CIO0);
int iosr_reg_value(int Reserved,int ADC,int BI,int TEMT,int THRE,int FE,
                      int OE,int DR,int DIO3,int DIO2,int DIO1,int DIO0,
                      int IO3,int IO2,int IO1,int IO0);
int tcr_reg_value(int Reserved,int FREE,int SOFT,int PSC,
                      int TRB,int TSS,int TDDR);
int wsgr_reg_value(int Reserved,int ISWS,int DSWS,int PSUWS,int PSLWS);

/* Name        Mem Offset */

#define CLK             0x0000
#define IC              0x0004
#define SDTR            0x0008
#define SSPCR           0x000C
#define ADTR            0x0010
#define ASPCR           0x0014
#define IOSR            0x0018
#define BRD             0x001C
#define TCR             0x0020
#define PRD             0x0024
#define TIM             0x0028
#define WSGR            0x002C

/* Each port is offset by 0004 in the assembly branch section
/* OUT...
   B...
   since in total both commands require 4 words of memory */
```

A.4 Functions Code ‘C209 : c209io.c

```
/************************************************
* Example C functions for IO registers for C209 *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
*************************************************/

/* The first function listed below is for sending
   a value to a register. The function then calls
   another function in assembly language */

extern int Port_Offset;
extern int Port_Value;

void port_function(int Reg_Offset, int Reg_Value)
{
       Port_Offset = Reg_Offset;       /* Sets Port Offset value */
       Port_Value = Reg_Value;         /* Sets Port value */

       out_to_port();                  /* Calls the ASM function */
}

/************************************************/

/* PRD TIM are counter registers that take a 16 bit word    */

/* The functions below return complete 16 bit values for
   the corresponding registers                          */

/************************************************/

#ifdef _TMS320C2XX

int tcr_reg_value(int Reserved,int PSC,int TRB,int TSS,int TDDR)
{
       int x = 0;
        x = ((Reserved<<12)|(PSC<<6)|(TRB<<5)|(TSS<<4)|(TDDR));
       return (x);
}
/************************************************/
int wsgr_reg_value(int Reserved,int AVIS,int ISWS,int DSWS,int PSWS)
{
       int x = 0;
        x = ((Reserved<<4)|(AVIS<<3)|(ISWS<<2)|(DSWS<<1)|(PSWS));
       return (x);
}
#endif
```

A.5 Header File ‘C209 : c209io.h

```
/************************************************
* TMS320C2xx Processor Header File C209         *
*           This file contains the necessary   *
*           Offsets for the out_port function  *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
************************************************/

/* Prototypes found in usercode.c */
extern  void out_to_port(void);    /* External asm function */
void port_function(int Reg_Offset, int Reg_Value);

/* Functions to set registers */
/* Functions can be found in c209io.lib */
int tcr_reg_value(int Reserved,int PSC,int TRB,int TSS,int TDDR);
int wsgr_reg_value(int Reserved,int AVIS,int ISWS,int DSWS,int PSWS);

/* Name        Mem Offset */

#define TCR             0x0000
#define PRD             0x0004
#define TIM             0x0008
#define WSGR            0x000C

/* Each port is offset by 0004 in the assembly branch section
/* OUT...
   B...
   since in total both commands require 4 words of memory */
```

Appendix B - Assembly Code Listings

B.1 Assembly Routine ‘x2xx : x2xxio.asm

```
; ************************************************
; * TMS320x2xx Processor Subroutine               *
; *           This function is called from C     *
; *                                               *
; * Written by Nicholas Holland                   *
; *           Texas Instruments                   *
; *           1996                                *
; *                                               *
; * Release Version 1.0                           *
; *                                               *
; *************************************************

        .def    _out_to_port            ; Function defined in ASM code
        .ref    _Port_Offset            ; Variable defined in C code
        .ref    _Port_Value             ; Variable defined in C code

; ******************************
; * Function called from C code *
; ******************************
        .text

_out_to_port:                           ; Function called from C
        POPD    *+                      ; Place return address on stack
        SAR     AR0,*+                  ; Store AR0 Frame Pointer
        SAR     AR1,*                   ; Store AR1 Stack Pointer

        ; Place code here using AR2 to access data if necessary

        LACC    #START                  ; Load ACC with Address of START
        ADD     _Port_Offset            ; Add Port Offset
        BACC                            ; Branche to correct OUT command

START:  OUT     _Port_Value,0FFE8h      ; CLK Port
        B       DONE                    ;
        OUT     _Port_Value,0FFECh      ; IC Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF0h      ; SDTR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF1h      ; SSPCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF4h      ; ADTR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF5h      ; ASPCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF6h      ; IOSR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF7h      ; BRD Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF8h      ; TCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF9h      ; PRD Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFAh      ; TIM Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFCh      ; WSGR Port
        B       DONE                    ;

        ; User code stops here
DONE:
        MAR     *,AR1                   ; Select AR1
        SBRK    #1                      ;
        LAR     AR0,*-                  ; Load AR0 from Stack
        PSHD    *                       ; Load Return Address from Stack
        RET                             ; Return from function call

; ******************************
```

B.2 Assembly Routine ‘C209 : c209io.asm

```
; ************************************************
; * TMS320C209 Processor Subroutine               *
; *           This function is called from C      *
; *                                               *
; * Written by Nicholas Holland                   *
; *           Texas Instruments                   *
; *           1996                                *
; *                                               *
; * Release Version 1.0                           *
; *                                               *
; *************************************************

        .def    _out_to_port            ; Function defined in ASM code
        .ref    _Port_Offset            ; Variable defined in C code
        .ref    _Port_Value             ; Variable defined in C code

; ******************************
; * Function called from C code *
; ******************************
        .text

_out_to_port:                           ; Function called from C
        POPD    *+                      ; Place return address on stack
        SAR     AR0,*+                  ; Store AR0 Frame Pointer
        SAR     AR1,*                   ; Store AR1 Stack Pointer

        ; Place code here using AR2 to access data if necessary

        LACC    #START                  ; Load ACC with Address of START
        ADD     _Port_Offset            ; Add Port Offset
        BACC                            ; Branche to correct OUT command

START:  OUT     _Port_Value,0FFFCh      ; TCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFDh      ; PRD Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFEh      ; TIM Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFFh      ; WSGR Port
        B       DONE                    ;

        ; User code stops here
DONE:
        MAR     *,AR1                   ; Select AR1
        SBRK    #1                      ;
        LAR     AR0,*-                  ; Load AR0 from Stack
        PSHD    *                       ; Load Return Address from Stack
        RET                             ; Return from function call

; ******************************
```

Appendix C - Run-Time Library Code Listings

C.1 Library Source File ‘C203 : c203io.src

```
!<arch>
x2xxio.h/       858268316   0     0     0       2205      ‘
/************************************************
* TMS320x2xx Processor Header File x2xx         *
*           This file contains the necessary   *
*           Offsets for the out_port function  *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
*************************************************/

/* Prototypes found in usercode.c */
extern  void out_to_port(void);    /* External asm function */
void port_function(int Reg_Offset, int Reg_Value);

/* Functions to set registers */
/* Functions can be found in x2xxio.lib */
int clk_reg_value(int Reserved,int CLKOUT1);
int ic_reg_value(int Reserved,int MODE,int FINT3,int FINT2,
                      int MINT3,int MINT2);
int sspcr_reg_value(int FREE,int SOFT,int TCOMP,int RFNE,int FT1,int FT0,
                      int FR1,int FR0,int IN1,int IN0,int XRST,int RRST,
                      int TXM,int MCM,int FSM,int DLD);
int aspcr_reg_value(int FREE,int SOFT,int URST,int Reserved,int DIM,int TIM,
                      int RIM,int STB,int CAD,int SETBRK,int CIO3,int CIO2,
                      int CIO1,int CIO0);
int iosr_reg_value(int Reserved,int ADC,int BI,int TEMT,int THRE,int FE,
                      int OE,int DR,int DIO3,int DIO2,int DIO1,int DIO0,
                      int IO3,int IO2,int IO1,int IO0);
int tcr_reg_value(int Reserved,int FREE,int SOFT,int PSC,
                      int TRB,int TSS,int TDDR);
int wsgr_reg_value(int Reserved,int ISWS,int DSWS,int PSUWS,int PSLWS);

/* Name        Mem Offset */

#define CLK             0x0000
#define IC              0x0004
#define SDTR            0x0008
#define SSPCR           0x000C
#define ADTR            0x0010
#define ASPCR           0x0014
#define IOSR            0x0018
#define BRD             0x001C
#define TCR             0x0020
#define PRD             0x0024
#define TIM             0x0028
#define WSGR            0x002C

/* Each port is offset by 0004 in the assembly branch section
/* OUT...
   B...
   since in total both commands require 4 words of memory */

x2xxio.c/       858268316   0     0     0       3570      ‘
/************************************************
* Example C functions for IO registers for x2xx *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
*************************************************/

/* The first function listed below is for sending
   a value to a register. The function then calls
   another function in assembly language */

extern int Port_Offset;
extern int Port_Value;

void port_function(int Reg_Offset, int Reg_Value)
{
       Port_Offset = Reg_Offset;       /* Sets Port Offset value */
       Port_Value = Reg_Value;         /* Sets Port value */

       out_to_port();                  /* Calls the ASM function */
}

/*************************************************/

/* SDTR ADDR are transmit registers that take a 16 bit word */
/* PRD TIM are counter registers that take a 16 bit word    */
/* BRD determines the baud rate and takes a 16 bit word     */

/* The functions below return complete 16 bit values for
   the corresponding registers                          */

/*************************************************/

#ifdef _TMS320C2XX

int clk_reg_value(int Reserved, int CLKOUT1)
{
       int x = 0;

       x = ((Reserved<<1)|(CLKOUT1));
       return (x);

}
/*************************************************/
int ic_reg_value(int Reserved,int MODE,int FINT3,int FINT2,int MINT3,
                      int MINT2)
{
       int x = 0;

       x = ((Reserved<<5)|(MODE<<4)|(FINT3<<3)|
                      (FINT2<<2)|(MINT3<<1)|(MINT2));
       return (x);
}
/*************************************************/
int sspcr_reg_value(int FREE,int SOFT,int TCOMP,int RFNE,int FT1,int FT0,
                      int FR1,int FR0,int IN1,int IN0,int XRST,int RRST,
                       int TXM,int MCM,int FSM,int DLB)

{
       int x = 0;
       x = ((FREE<<15)|(SOFT<<14)|(TCOMP<<13)|(RFNE<<12)|(FT1<<11)|
                      (FT0<<10)|(FR1<<9)|(FR0<<8)|(IN1<<7)|(IN0<<6)|
                       (XRST<<5)|(RRST<<4)|(TXM<<3)|(MCM<<2)|(FSM<<1)|(DLB));
       return (x);
}
/*************************************************/
int aspcr_reg_value(int FREE,int SOFT,int URST,int Reserved,int DIM,int TIM,
                      int RIM,int STB,int CAD,int SETBRK,int CIO3,int CIO2,
                      int CIO1,int CIO0)
{
       int x = 0;
       x = ((FREE<<15)|(SOFT<<14)|(URST<<13)|(Reserved<<10)|(DIM<<9)|
                      (TIM<<8)|(RIM<<7)|(STB<<6)|(CAD<<5)|(SETBRK<<4)|
                      (CIO3<<3)|(CIO2<<2)|(CIO1<<1)|(CIO0));
       return (x);
}
/*************************************************/
int iosr_reg_value(int Reserved,int ADC,int BI,int TEMT,int THRE,int FE,
                      int OE,int DR,int DIO3,int DIO2,int DIO1,int DIO0,
                      int IO3,int IO2,int IO1,int IO0)
{
       int x = 0;
       x = ((Reserved<<15)|(ADC<<14)|(BI<<13)|(TEMT<<12)|(THRE<<11)|
                      (FE<<10)|(OE<<9)|(DR<<8)|(DIO3<<7)|(DIO2<<6)|
                      (DIO1<<5)|(DIO0<<4)|(IO3<<3)|(IO2<<2)|(IO1<<1)|(IO0));
       return (x);
}
/*************************************************/
int tcr_reg_value(int Reserved,int FREE,int SOFT,int PSC,
                      int TRB,int TSS,int TDDR)
{
       int x = 0;
        x = ((Reserved<<12)|(FREE<<11)|(SOFT<<10)|(PSC<<6)|(TRB<<5)|
                       (TSS<<4)|(TDDR));
       return (x);
}
/*************************************************/
int wsgr_reg_value(int Reserved,int ISWS,int DSWS,int PSUWS,int PSLWS)
{
       int x = 0;
       x = ((Reserved<<12)|(ISWS<<9)|(DSWS<<6)|
                      (PSUWS<<3)|(PSLWS));
       return (x);
}
#endif
x2xxio.asm/     858268384   0     0     0       3040      ‘
; ************************************************
; * TMS320x2xx Processor Subroutine              *
; *           This function is called from C     *
; *                                              *
; * Written by Nicholas Holland                  *
; *           Texas Instruments                  *
; *           1996                               *
; *                                              *
; * Release Version 1.0                          *
; *                                              *
; ************************************************

        .def    _out_to_port            ; Function defined in ASM code
        .ref    _Port_Offset            ; Variable defined in C code
        .ref    _Port_Value             ; Variable defined in C code

; ******************************
; * Function called from C code *
; ******************************
        .text

_out_to_port:                           ; Function called from C
        POPD    *+                      ; Place return address on stack
        SAR     AR0,*+                  ; Store AR0 Frame Pointer
        SAR     AR1,*                   ; Store AR1 Stack Pointer

        ; Place code here using AR2 to access data if necessary

        LACC    #START                  ; Load ACC with Address of START
        ADD     _Port_Offset            ; Add Port Offset
        BACC                            ; Branche to correct OUT command

START:  OUT     _Port_Value,0FFE8h      ; CLK Port
        B       DONE                    ;
        OUT     _Port_Value,0FFECh      ; IC Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF0h      ; SDTR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF1h      ; SSPCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF4h      ; ADTR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF5h      ; ASPCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF6h      ; IOSR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF7h      ; BRD Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF8h      ; TCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFF9h      ; PRD Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFAh      ; TIM Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFCh      ; WSGR Port
        B       DONE                    ;
        ; User code stops here
DONE:
        MAR     *,AR1                   ; Select AR1
        SBRK    #1                      ;
        LAR     AR0,*-                  ; Load AR0 from Stack
        PSHD    *                       ; Load Return Address from Stack
        RET                             ; Return from function call

; ******************************
```

C.2 Library Source File ‘C209 : c209io.src

```
!<arch>
c209io.h/       858268486   0     0     0       1296      ‘
/************************************************
* TMS320C209 Processor Header File C209         *
*           This file contains the necessary    *
*           Offsets for the out_port function   *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
*************************************************/

/* Prototypes found in usercode.c */
extern  void out_to_port(void);    /* External asm function */
void port_function(int Reg_Offset, int Reg_Value);

/* Functions to set registers */
/* Functions can be found in c209io.lib */
int tcr_reg_value(int Reserved,int PSC,int TRB,int TSS,int TDDR);
int wsgr_reg_value(int Reserved,int AVIS,int ISWS,int DSWS,int PSWS);

/* Name        Mem Offset */

#define TCR             0x0000
#define PRD             0x0004
#define TIM             0x0008
#define WSGR            0x000C

/* Each port is offset by 0004 in the assembly branch section
/* OUT...
   B...
   since in total both commands require 4 words of memory */
c209io.c/       847901112   0     0     0       1650      ‘
/************************************************
* Example C functions for IO registers for C209 *
*                                               *
* Written by Nicholas Holland                   *
*           Texas Instruments                   *
*           1996                                *
*                                               *
* Release Version 1.0                           *
*                                               *
*************************************************/

/* The first function listed below is for sending
   a value to a register. The function then calls
   another function in assembly language */

extern int Port_Offset;
extern int Port_Value;

void port_function(int Reg_Offset, int Reg_Value)
{
       Port_Offset = Reg_Offset;       /* Sets Port Offset value */
       Port_Value = Reg_Value;         /* Sets Port value */

       out_to_port();                  /* Calls the ASM function */
}

/************************************************/
```

```
/* PRD TIM are counter registers that take a 16 bit word    */

/* The functions below return complete 16 bit values for
   the corresponding registers                          */

/*************************************************/

#ifdef _TMS320C2XX

int tcr_reg_value(int Reserved,int PSC,int TRB,int TSS,int TDDR)
{
       int x = 0;
        x = ((Reserved<<12)|(PSC<<6)|(TRB<<5)|(TSS<<4)|(TDDR));
       return (x);
}
/*************************************************/
int wsgr_reg_value(int Reserved,int AVIS,int ISWS,int DSWS,int PSWS)
{
       int x = 0;
        x = ((Reserved<<4)|(AVIS<<3)|(ISWS<<2)|(DSWS<<1)|(PSWS));
       return (x);
}
#endif
c209io.asm/     858268484   0     0     0       2286      ‘
; ************************************************
; * TMS320C209 Processor Subroutine              *
; *           This function is called from C     *
; *                                              *
; * Written by Nicholas Holland                  *
; *           Texas Instruments                  *
; *           1996                               *
; *                                              *
; * Release Version 1.0                          *
; *                                              *
; ************************************************

        .def    _out_to_port            ; Function defined in ASM code
        .ref    _Port_Offset            ; Variable defined in C code
        .ref    _Port_Value             ; Variable defined in C code

; ******************************
; * Function called from C code *
; ******************************
        .text

_out_to_port:                           ; Function called from C
        POPD    *+                      ; Place return address on stack
        SAR     AR0,*+                  ; Store AR0 Frame Pointer
        SAR     AR1,*                   ; Store AR1 Stack Pointer

        ; Place code here using AR2 to access data if necessary

        LACC    #START                  ; Load ACC with Address of START
        ADD     _Port_Offset            ; Add Port Offset
        BACC                            ; Branche to correct OUT command

START:  OUT     _Port_Value,0FFFCh      ; TCR Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFDh      ; PRD Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFEh      ; TIM Port
        B       DONE                    ;
        OUT     _Port_Value,0FFFFh      ; WSGR Port
        B       DONE                    ;

        ; User code stops here
DONE:
        MAR     *,AR1                   ; Select AR1
        SBRK    #1                      ;
        LAR     AR0,*-                  ; Load AR0 from Stack
        PSHD    *                       ; Load Return Address from Stack
        RET                             ; Return from function call

; ******************************
```

Appendix D - Setup Files

D.1 Source Code Command file userc2xx.cmd

```
usercode.obj
-o user.out
-m user.map
-c
-stack 20h
-l c:\users\nic\fixed\v660\lib\rts2xx.lib
-l x2xxio.lib   /* The library file must be changed according to x2xx */

/*-----------------------------------------------------------------------*/
/*  MEMORY SPECIFICATION                                                  */
/*  Block B0 is configured as data memory (CNF=0) and MP/MC- = 0          */
/*  (microcomputer  mode).RAM=1 OVLY=1                                    */
/*-----------------------------------------------------------------------*/

MEMORY
{
      PAGE 0:    PROG:   org = 0000h, length = 0800h

      PAGE 1:    B2  :   org = 0060h, length = 0020h
                 B0  :   org = 0200h, length = 0100h
                 B1  :   org = 0300h, length = 0100h
                 DATA:   org = 0800h, length = 0400h

      PAGE 2 :  IOMEM:   org = 0FFE8h, length = 0017h
}

SECTIONS
{
        .text   >       PROG    PAGE 0
        .cinit  >       PROG    PAGE 0

        .bss    >       DATA    PAGE 1
        .const  >       B2      PAGE 1
        .stack  >       B0      PAGE 1

        .ioregs >       IOMEM   PAGE 2
}
```

D.2 Linker file : userc2xx.bat

```
@echo off
rem Batch file to assemble and link the x2xxio program
rem
rem Written by Nicholas Holland
rem            Texas Instruments Inc.
rem            1996
rem
rem Release version 1.0

dspcl  -v2xx usercode.c

dsplnk userc2xx.cmd
```

D.3 Simulator Command file : simuser.cmd

```
; Configuration for the x2xx

; Program space

ma 0x0000 , 0 , 0x0800 , RAM|EX|R|W  ; Prog memory
ma 0x0C00 , 0 , 0x0100 , RAM|EX      ; Prog memory for asm function

; Data Space

ma 0x0060 , 1 , 0x0020 , RAM|EX  ;  Data memory
ma 0x0200 , 1 , 0x00FF , RAM|EX  ;  Data memory
ma 0x0300 , 1 , 0x00FF , RAM|EX  ;  Data memory
ma 0x0800 , 1 , 0x0400 , RAM|EX  ;  Data memory
```

```
; IO Space

ma 0x0FFE8,2,1,OPORT
ma 0x0FFEC,2,1,OPORT
ma 0x0FFF0,2,1,OPORT
ma 0x0FFF1,2,1,OPORT
ma 0x0FFF4,2,1,OPORT
ma 0x0FFF5,2,1,OPORT
ma 0x0FFF6,2,1,OPORT
ma 0x0FFF7,2,1,OPORT
ma 0x0FFF8,2,1,OPORT
ma 0x0FFF9,2,1,OPORT
ma 0x0FFFA,2,1,OPORT
ma 0x0FFFC,2,1,OPORT

mc 0x0FFE8,2,1,val1,write
mc 0x0FFEC,2,1,val2,write
mc 0x0FFF0,2,1,val3,write
mc 0x0FFF1,2,1,val4,write
mc 0x0FFF4,2,1,val5,write
mc 0x0FFF5,2,1,val6,write
mc 0x0FFF6,2,1,val7,write
mc 0x0FFF7,2,1,val8,write
mc 0x0FFF8,2,1,val9,write
mc 0x0FFF9,2,1,valA,write
mc 0x0FFFA,2,1,valB,write
mc 0x0FFFC,2,1,valC,write
```

D.4 Simulator log file : userinit.log

```
mem 0x0200
mix
go main

wa ar0
wa ar1
wa ar2
wa ar3
wa st0>>13,arp
win WATCH
size 20,6
move 60,14

WIN CPU
SIZE 25,15
MOVE 55,1

WIN MEMORY
SIZE 35,6
MOVE 45,19

win WATCH
```