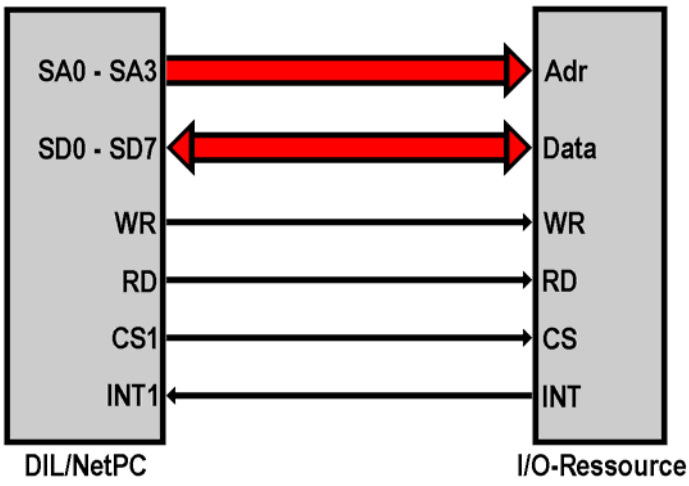
How to program the DNP/1110 Expansion Bus

- ***1. Step***: There are 8 data bits, 4 address bits and some control lines. Design the interface between your external I/O resources and the DNP/1110. Use the 4 chip select lines of the DNP/1110.



- ***2. Step***: Write your own application code. Use the following code for a sample to access external I/O resources from a C program, which runs in the Linux user space.

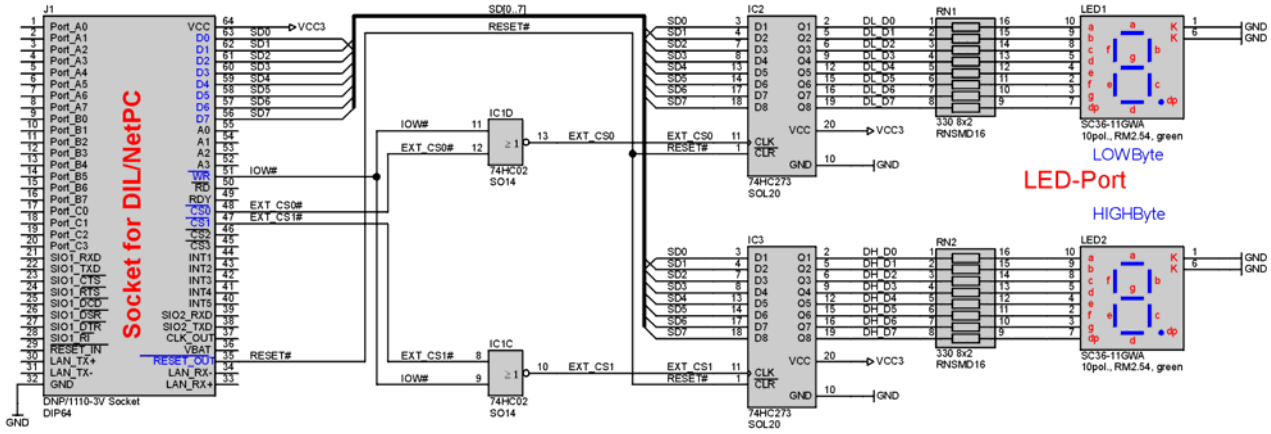
```
// Demo "7 segment display" access for DNP/1110-3V
// Written by MHA - 22.01.2002
// mmap function

#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <ctype.h>
#include <sys/mman.h>

#define CPUTYPE "DNP/1110-3V"
#define VERSION "20020122"

typedef unsigned char  u8;
typedef unsigned short u16;
typedef unsigned int   u32;

void *mapmemory(off_t phy_addr, size_t phy_lenght);
u8 chartoseg(u8 value);

// function main…
// *******************************************************

int main (void)
{
    int iLow, iHigh;

    // we have 8 bit chipselects
```

```
volatile u8 *ptr_CS1, *ptr_CS2;

// clear screen

printf("7 segment display demo\n");

// check user identity

if (geteuid() != 0) {
   printf("No root access rights !\n");
   exit(1);
}

// DNP/1110-3V CS0 @ 0x30000000

printf("Mapping CS1 into userspace memory... ");
if ((ptr_CS1 = mapmemory(0x30000000,0x1000)) == NULL)
   exit(1);
printf("done.\n");

// DNP/1110-3V CS1 @ 0x31000000

printf("Mapping CS2 into userspace memory... ");
if ((ptr_CS2 = mapmemory(0x31000000,0x1000)) == NULL)
   exit(1);
printf("done.\n\n");

// display "00", "01," -> "FF"

for (iHigh = '0'; iHigh <= 'F'; iHigh++)
   for (iLow = '0'; iLow <= 'F'; iLow++) {

      if (iLow == ':')
         iLow += 7;              //jump over ':' - '@'
      if (iHigh == ':')
         iHigh += 7;             //jump over ':' - '@'

      // display low character (write to CS0)

      *ptr_CS1 = chartoseg(iLow);

      // display high character (write to CS1)

      *ptr_CS2 = chartoseg(iHigh);

      printf("\rDisplay : %c%c", iHigh, iLow);
      fflush(stdout);
      sleep(1);
   }

// display low character (write to CS0)

*ptr_CS1 = chartoseg('.');
```

```
    // display high character (write to CS1)

    *ptr_CS2 = chartoseg(' ');

    printf("\rDisplay :  .\n");
    return 0;
}

// function chartoseg…
// *******************************************************

u8 chartoseg(u8 value)
{
    switch(toupper(value))
    {
       case '0': return(0x3f);   // display '0'
       case '1': return(0x06);   // display '1'
       case '2': return(0x5b);   // display '2'
       case '3': return(0x4f);   // display '3'
       case '4': return(0x66);   // display '4'
       case '5': return(0x6d);   // display '5'
       case '6': return(0x7d);   // display '6'
       case '7': return(0x07);   // display '7'
       case '8': return(0x7f);   // display '8'
       case '9': return(0x6f);   // display '9'
       case 'A': return(0x77);   // display 'A'
       case 'B': return(0x7c);   // display 'B'
       case 'C': return(0x58);   // display 'C'
       case 'D': return(0x5e);   // display 'D'
       case 'E': return(0x79);   // display 'E'
       case 'F': return(0x71);   // display 'F'
       case '.': return(0x80);   // display '.'
       case ' ': return(0x00);   // display ' '
    }
    return(0x00);                // display ' '
}

// function mapmemory…
// *******************************************************

void *mapmemory(off_t phy_addr, size_t phy_lenght)
{
#define MAP_PAGESIZE 4096UL

int iFd;
void *pMem;

    if ((phy_addr % MAP_PAGESIZE) != 0) {
       fprintf(stderr,"physical address error!\n");
       return(NULL);
    }

    if ((phy_lenght % MAP_PAGESIZE) != 0) {
       fprintf(stderr,"physical lenght error!\n");
       return(NULL);
```

```
    }

    // open mem device for read/write

    iFd = open("/dev/mem", O_RDWR | O_SYNC);
    if (iFd < 0) {
       fprintf(stderr,"open of /dev/mem fail !\n");
       return(NULL);
    }

    // get pointer to DNP1110 memory

    pMem = mmap(NULL,
                phy_lenght,
                (PROT_READ | PROT_WRITE),
                MAP_SHARED,
                iFd,
                phy_addr);

    if ((pMem == MAP_FAILED) || (pMem == NULL)) {
       fprintf(stderr,"mmap of /dev/mem fail !\n");
       return(NULL);
    }

    // close mem device

    if (close(iFd) != 0)
       fprintf(stderr,"close of /dev/mem fail !\n");

    return(pMem);
}
```

- ***3. Step***: Understand the sample code. The sample is using 2 external 8-bit output ports (write-only). One port is driven by chip select CS0, the other by CS1. Each 8-bit output port drives a seven-segment LED display unit.



Appendix 1: DNP/1110 Pinout - JEDEC 64-pin DIL Connector (1. Part)

Pin	*Name*	*Group*	*Function*
1	PA0	PIO	Parallel I/O, Port A, Bit 0
2	PA1	PIO	Parallel I/O, Port A, Bit 1
3	PA2	PIO	Parallel I/O, Port A, Bit 2
4	PA3	PIO	Parallel I/O, Port A, Bit 3
5	PA4	PIO	Parallel I/O, Port A, Bit 4
6	PA5	PIO	Parallel I/O, Port A, Bit 5
7	PA6	PIO	Parallel I/O, Port A, Bit 6
8	PA7	PIO	Parallel I/O, Port A, Bit 7
9	PB0	PIO	Parallel I/O, Port B, Bit 0
10	PB1	PIO	Parallel I/O, Port B, Bit 1
11	PB2	PIO	Parallel I/O, Port B, Bit 2
12	PB3	PIO	Parallel I/O, Port B, Bit 3
13	PB4	PIO	Parallel I/O, Port B, Bit 4
14	PB5	PIO	Parallel I/O, Port B, Bit 5
15	PB6	PIO	Parallel I/O, Port B, Bit 6
16	PB7	PIO	Parallel I/O, Port B, Bit 7
17	PC0	PIO	Parallel I/O, Port C, Bit 0
18	PC1	PIO	Parallel I/O, Port C, Bit 1
19	PC2	PIO	Parallel I/O, Port C, Bit 2
20	PC3	PIO	Parallel I/O, Port C, Bit 3
21	RXD1	SIO	COM1 Serial Port, RXD Pin
22	TXD1	SIO	COM1 Serial Port, TXD Pin
23	CTS1	SIO	COM1 Serial Port, CTS Pin
24	RTS1	SIO	COM1 Serial Port, RTS Pin
25	DCD1	SIO	COM1 Serial Port, DCD Pin
26	DSR1	SIO	COM1 Serial Port, DSR Pin
27	DTR1	SIO	COM1 Serial Port, DTR Pin
28	RI	SIO	COM1 Serial Port, RI Pin
29	RESIN	RESET	RESET Input
30	TX+	LAN	Ethernet Interface, TX+ Pin
31	TX-	LAN	Ethernet Interface, TX- Pin
32	GND	----	Ground

Table 1: DNP/1110 Pinout - Pin 1 to 32

Appendix 2: DNP/1110 Pinout - JEDEC 64-pin DIL Connector (2. Part)

Pin	*Name*	*Group*	*Function*
33	RX+	LAN	Ethernet Interface, RX+ Pin
34	RX-	LAN	Ethernet Interface, RX- Pin
35	RESOUT	RESET	RESET Output
36	VBAT	PSP	Real Time Clock Battery Input
37	CLKOUT	PSP	Clock Output (Default 3.6864 MHz)
38	TXD2	PSP	COM2 Serial Port, TXD Pin
39	RXD2	PSP	COM2 Serial Port, RXD Pin
40	**INT5**	**PSP**	**Interrupt Input 5**
41	**INT4**	**PSP**	**Interrupt Input 4**
42	**INT3**	**PSP**	**Interrupt Input 3**
43	**INT2**	**PSP**	**Interrupt Input 2**
44	**INT1**	**PSP**	**Interrupt Input 1**
45	**CS4**	**PSP**	**Chip Select Output 4**
46	**CS3**	**PSP**	**Chip Select Output 3**
47	**CS2**	**PSP**	**Chip Select Output 2**
48	**CS1**	**PSP**	**Chip Select Output 1**
49	**RDY**	**PSP**	**External Ready Input**
50	**RD**	**PSP**	**Read Signal, Expansion Bus**
51	**WR**	**PSP**	**Write Signal, Expansion Bus**
52	**SA3**	**PSP**	**Expansion Bus, Address Bit 3**
53	**SA2**	**PSP**	**Expansion Bus, Address Bit 2**
54	**SA1**	**PSP**	**Expansion Bus, Address Bit 1**
55	**SA0**	**PSP**	**Expansion Bus, Address Bit 0**
56	**SD7**	**PSP**	**Expansion Bus, Data Bit 7**
57	**SD6**	**PSP**	**Expansion Bus, Data Bit 6**
58	**SD5**	**PSP**	**Expansion Bus, Data Bit 5**
59	**SD4**	**PSP**	**Expansion Bus, Data Bit 4**
60	**SD3**	**PSP**	**Expansion Bus, Data Bit 3**
61	**SD2**	**PSP**	**Expansion Bus, Data Bit 2**
62	**SD1**	**PSP**	**Expansion Bus, Data Bit 1**
63	**SD0**	**PSP**	**Expansion Bus, Data Bit 0**
64	VCC	----	3.3 Volt Power Input

Table 2: DNP/1110 Pinout - Pin 33 to 64

Appendix 3: DNP/1110 Memory Map

Physical Addr.	Virtual Addr.	Description	Cached	Buffered	Access
0x00000000-0x07FFFFF	0xE8000000-0xEFFFFFFF	16 MByte FLASH	No	No	R/W
0x20000000-0x20FFFFF	0xF6000000-0xF6FFFFF	Ethernet Controller	No	No	R/W
0x30000000-0x30FFFFF	**None**	**Chip Select Signal CS0**	**No**	**No**	**R/W**
0x31000000-0x31FFFFF	**None**	**Chip Select Signal CS1**	**No**	**No**	**R/W**
0x32000000-0x32FFFFF	**None**	**Chip Select Signal CS2**	**No**	**No**	**R/W**
0x33000000-0x33FFFFF	**None**	**Chip Select Signal CS3**	**No**	**No**	**R/W**
0x80000000-0xB7FFFFF	0x80000000-0xB7FFFFF	SA-1110 internal Registers	No	No	R/W
0xC0000000-0xC7FFFFF	0x00000000-0x07FFFFF	32 MByte SDRAM	Yes	Yes	R/W

Table 3: DNP/1110 Memory Map