Parallel Ports and The IBM PC
----------------------------
This document details the parallel printer ports of the IBM-PC. Each of the printer ports conforms to a modified Centronics Parallel Interface. The connection to the PC is through a DB25 25 pin connector. The IBM-PC will support up to three parallel ports. Each port provides 12 TTL outputs and 5 TTL inputs. This document contains the following data:

a. DB25 PINOUT, and signal names where known.
b. PORT addresses of each parallel port.
c. DATA, CONTROL, and INPUT PORTS pinouts with bit positions within each byte.
d. Example programs written in C highlighting input and output to the ports.
e. Clock display circuit diagram with parts list.
f. Closing comment.

DB25 PINOUT

```
- STROBE -------> 1
+D0 -------> 2
+D1 -------> 3
+D2 -------> 4
+D3 -------> 5
+D4 -------> 6
+D5 -------> 7
+D6 -------> 8
+D7 -------> 9
- ACK <------ 10
+BUSY <------ 11
+PE <------ 12
+SEL <------ 13

14 <------- AUTO FEED
15 -------> ERROR
16 <------ INITIALIZE
17 <------ SELECT INPUT
18 <--- DB0
19 | G | DB1
20 | R | DB2
21 | O | DB3
22 | U | DB4
23 | N | DB5
24 | D | DB6
25 | H | DB7

Output Signal -------> G
(From Computer) O
(To Computer) D

Pins 1, 11, 14, 17 are negative TTL, meaning
0 is ON & 1 is OFF
```
PARALLEL Port Addresses    DATA PORT (Output from computer)

LPT1: 956     3BC Hex
LPT2: 888     378 Hex
LPT3: 632     278 Hex

PIN #      9       8      7      6      5       4      3       2
BIT #      7       6      5      4      3       2      1       0
VALUE     128      64     32     16      8       4      2       1

PARALLEL Port Addresses    CONTROL PORT (Output from computer)

LPT1: 958     3BE Hex
LPT2: 890     37A Hex
LPT3: 634     27A Hex

PIN #      -       -      -      -     17      16     14       1
BIT #      -       -      -      -      3       2      1       0
VALUE     -       -      -      -      8       4      2       1

Notes:
1. -? denotes negative TTL signal, name unknown.
2. INIT is INITIAL.
3. -AF is negative TTL signal, AUTOFEED is signal name.
4. -ST is negative TTL signal, STROBE is signal name.
5. NA not applicable.
PARALLEL Port Addresses  INPUT PORT (Input to computer)

<table>
<thead>
<tr>
<th>Port</th>
<th>Address</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPT1</td>
<td>957</td>
<td>3BD</td>
</tr>
<tr>
<td>LPT2</td>
<td>889</td>
<td>379</td>
</tr>
<tr>
<td>LPT3</td>
<td>633</td>
<td>279</td>
</tr>
</tbody>
</table>

Note: LPT is used for printers.


<table>
<thead>
<tr>
<th>PIN #</th>
<th>11</th>
<th>10</th>
<th>12</th>
<th>13</th>
<th>15</th>
<th>-</th>
<th>-</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT #</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VALUE</td>
<td>128</td>
<td>64</td>
<td>32</td>
<td>16</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Notes:

1. ? signal name unknown.
2. NA not applicable.
3. -BUSY is negative TTL signal.
Example of Negative TTL logic

<table>
<thead>
<tr>
<th>PIN #</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT #</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>VALUE</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

DATA PORT

| 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 |
|----------------------------------|
| INPUT PORT                       |

| 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 |

Notes:

1. A 1 bit in the DATA port causes 2.5 to 5.0 volts to appear on the corresponding pin on the DB25 connector. In the example above PIN # 6 is on.

2. BIT # 7 of the INPUT port has negative TTL logic, where 2.5 to 5.0 volts on PIN # 11 is logic level zero. The other bits of the INPUT port are normal TTL logic.
/* CLOCK.C ==> This program runs the binary clock */

#include "coni.o.h"
#define PORT HR 634
#define PORT M N 632
#define OFF M N 0
#define OFF HRS 0x0b
#define WN K 64 /* bit 7 of minute byte used for seconds */

main()
{
    unsigned char ctr m in, ctr hr, status, hours, minutes, seconds, 
          hundrds, lastsec, second_on();
    void time(), turn on(), turn off();

    /* start by turning off all circuits */
    ctr m in = OFF M N;
    ctr hr = OFF HRS;

    outp (PORT HR, ctr hr);
    outp (PORT M N, ctr m in);

    /* the clock will run indefinitely */
    for (;;) {
        /* get the time */
        time(&hours, &minutes, &seconds, &hundrds);

        /* the hours require special handling, DOS returns hours in 24 */
        /* hour format. Convert to 12 hr format. Next XOR hours against */
        /* OFF_HRS to get the proper bit pattern for the hours. */
        ctr hr = (((hours = hours % 12) ? hours : 12) ^ OFF_HRS;

        /* the minutes are easy, output ASI S */
        ctr m in = minutes;

        /* Display the hours and minutes */
        outp (PORT HR, ctr hr);
        outp (PORT M N, ctr m in);

        /* wink the seconds, up to the next minute */
for (lastsec = seconds; seconds <= 59; ) {
    time (&hours, &minutes, &seconds, &hundreds);
    if (lastsec != seconds) { /* seconds has changed */
        lastsec = seconds;
        /* if second is on turn off, else turn on */
        if (!second_on(ctr_min, WNK))
            turn_on (PORT_MIN, &ctr_min);
        else
            turn_off (PORT_MIN, &ctr_min);
    }
    /* seconds up to 59, exit loop */
    if (seconds == 59)
        break;
}

unsigned char second_on(control, sec)
unsigned char control, sec;
{
    return (control & sec);
}

void turn_on (port, control)
unsigned char *control;
int port;
{
    *control = *control | WNK;
    outp (port, *control);
}

void turn_off (port, control)
unsigned char *control;
int port;
{
    *control = *control & ~WNK;
    outp (port, *control);
}
/* PORTTST.C -- This program is used to determine if the information I have gathered about Input Ports is correct. The program is to send pre-determined values to PORT 632 bits 0-4 */
/* the bits correspond to pins 2-6 of a printer parallel port. */
/* Pins 2-6 are then connected to Pins 15, 13, 12, 10, 11 respectively. Pins 10-13, 15 correspond to Bits 7-3 of Input PORT */
/* 633. For a given input a pre-determined set of values will */
/* be returned. Pin 11 of the Input port is negative TTL */

#define PORTE 633
#define PORTD 632
#include "conio.h"

main()
{
    static char test_values[] = {16, 17, 18, 20, 24, 0};

    /* test_value => 16 return 0 */
    /* test_value => 17 return 1 */
    /* test_value => 18 return 2 */
    /* test_value => 20 return 4 */
    /* test_value => 24 return 8 */
    /* test_value => 0 return 16 */

    unsigned char bit, send;
    void porttst();

    /* turn off all bits in send */
    for (bit = 0; bit < sizeof(test_values); bit++) {
        porttst(PORTD, PORTE, *(test_values + bit));
    }
}

void porttst(pd, pe, send)
int pd, pe;
unsigned char send;
{
    unsigned char result;
    int bits();

    outp(pd, send);
    result = inp(pe);

    /* display the bit patterns sent and received */
    printf("\nOut Value ");
    bits(4, 0, send);
    printf(" In Value ");
    bits(7, 3, result);
}
OUTPUT from PORTTST.C

C>porttst

Out Value 10000 In Value 00000
Out Value 10001 In Value 00001
Out Value 10010 In Value 00010
Out Value 10100 In Value 00100
Out Value 11000 In Value 01000
Out Value 00000 In Value 10000
/* DTIME.C ==> This program generates binary code representing  */
/* 0 to 12, which are then output from the computer's control port */
/* (actually part of the parallel printer port) to 4 LEDs that    */
/* are connected to pins 1, 14, 16 & 17 of the parallel port      */
/* with the first being binary digit 0 and the latter being digit */
/* 3.                                                            */
/* See diagram of the Control Port for more data.                */

#define CONTROL_PORT   634
#define XOR           0xb

main()
{
    char count, send;
    int bits();

    printf("\nCOUNT\tHOUR\t\tXOR\t\tACTUAL BITS\n");

    for (count = 0; count <= 12; count++) {
        send = count ^ XOR;
        outp(CONTROL_PORT, send);

        /* in practice the following is not used. it exists */
        /* for illustration purposes only.                    */
        printf("\n%4d\t", count);
        bits (3, 0, count);

        printf("\n%4d\t", count);
        bits (3, 0, XOR);

        printf("\n%4d\t", count);
        bits (3, 0, send);
    }
}

COUNT  HOUR  XOR  ACTUAL BITS
0  0000  1011  1011
1  0001  1011  1010
2  0010  1011  1001
3  0011  1011  1000
4  0100  1011  1111
5  0101  1011  1110
6  0110  1011  1101
7  0111  1011  1100
8  1000  1011  0011
9  1001  1011  0010
10  1010  1011  0001
11  1011  1011  0000
12  1100  1011  0111
CLOCK CIRCUIT

Minutes Display (uses DATA PORT)

PIN #    R1     D1
2        o- - - - \ \ / - - - - | <= - - - - +
          |    <--- Ground uses pins
          R2     D2
3        o- - - - \ \ / - - - - | <= - - - - |
          |          18 - 25.
          R3     D3
4        o- - - - \ \ / - - - - | <= - - - - |
          |   Parts List: R1 - R10: 220\(\Omega\) resistors
          R4     D4 |   D1 - D10: 2.5vdc @20ma LED's
5        o- - - - \ \ / - - - - | <= - - - - |
          |   Misc: R5     D5
          |   1 - DB25 male connector
          |   snap on is best.
          R6     D6
6        o- - - - \ \ / - - - - | <= - - - - |
          1 - 25 line ribbon cable
          R7     D7
7        o- - - - \ \ / - - - - | <= - - - - |

Hours Display (uses CONTROL PORT)

PIN #    R7     D7
1        o- - - - \ \ / - - - - | <= - - - - +
          |    <--- Ground uses pins
          R8     D8
14       o- - - - \ \ / - - - - | <= - - - - |
          |          18 - 25.
          R9     D9
16       o- - - - \ \ / - - - - | <= - - - - |
          |          R10  D10
17       o- - - - \ \ / - - - - | <= - - - - |