I²C bus expander AN036

INTRODUCTION

This application note describes two PLD designs made with the PLC42VA12. Both designs are controller functions for an I²C-bus n-bit I/O expander. The first design is a controller function for a n-bit I²C-bus Input Expansion (I²C-bus Slave Transmitter function) and the second one for a n-bit I²C-bus Output Expansion (I²C-bus Slave Receiver function).

The I²C-bus is a 2-way, 2-line communication between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). The designs provide remote input or output expansion for our Philips micro controller families via the two-line serial bidirectional I²C-Bus. The I²C-Bus slave address of the designs is equal to the address of the PCF8574 (remote 8-bit I/O expander). The I²C-Bus has been specified for 100kHz, but the PLC42VA12 designs can go up to 1MHz. This makes the designs suitable as test vehicle for the new fast I²C-Bus standard of 400 kHz.

The PLC42VA12 is the most powerfull PLD device in a DIL-package of Philips Semiconductors. The designs use almost all resources and most of the features of the PLC42VA12 e.g. combination of synchronous and asynchronous logic, 3-State outputs used as open collector outputs and a combination of output flip-flops and state flip-flops.

The design has been verified on a bread-board. This board contains the two PLC42VA12 controller designs, four 74HC165 devices, four 74HC595 devices and all the circuitry necessary to read 32 DIP-switches and to control 32 LED's.

This application note gives first a general description of the designs. Then it describes the characteristics of the I²C-Bus and some basic functions (tricks) used in both designs. You will find a detailed description of the designs and the I²C-Bus protocols of the controller functions in the sections entitled, I²C-Bus Slave Transmitter Function and I²C-Bus Slave Receiver Function. The appendix, gives all the used design files in the SNAP syntax. SNAP is the Philips Semiconductors PLD design software package. You will find the equation entry files EQN, the simulation control files SCL and the pinning files PIN. The last sheet gives the schematic diagram of the bread-board.

GENERAL DESCRIPTION OF THE DESIGNS

The two designs described in this report are both controller functions for an I²C-Bus n-bit I/O-expander. The designs were made on a customer request to have a solution for his problem to address via the I²C-Bus more than 200 bits of inputs and outputs. With the existing I²C-Bus devices the maximum number of input and/or output bits is 16 x 8 = 128 (8 x PCF8574 + 8 x PCF8574A). The designs work fully according to the I²C-Bus specification at 100kHz.

When an I²C-Bus master device (e.g. a micro controller) has to read data from or write data to the remote I/O-expander devices PCF8574 and PCF8574A, it first sends the I²C-Bus slave address of the device and then reads or writes only one byte. The two controller functions, the Slave-Receiver and the Slave Transmitter, don t have this problem. For these designs, the master sends the slave address only once, and then reads or writes one or multiple data-bytes. The master device, decides the number of bytes. The slave addresses used for the designs are identical to the slave addresses of the PCF8574 and PCF8574A devices.

The I²C-bus has been specified for a 100 kHz clock (SCL). With the internal maximum system clock of 8 MHz, the two PLC42VA12 designs can go upto an I²C-bus clock of 1MHz. This makes the design suitable as test vehicle for the new fast I²C-bus standard of 400 kbits/s. The speed is the only additional specification point of this new I²C-bus specification that can be met. The other new specification points as Schmitt-trigger inputs and slope control of the falling edges of the SDA and SCL signals can not be met.

The PLC42VA12 has been chosen, because of its special hardware features. These features are not available in other PLD devices available in a 24-pin DIL package e.g. the PL22V10. Some of the used features are:

- Combination of synchronous and asynchronous logic.
- Combination of D-type flip-flops with JK-type flip-flops.
- Flip-flops used as state registers. The M-pins used as inputs and/or outputs.
- 3-State buffers used as open collector outputs. The 3-State control-input of an I/O Output buffer is used as logic input. The input of that buffer is connected to the ground.

The intention was, to put both the Slave Transmitter and the Slave Receiver controller in one device. Unfortunately, the resources of the PLC42VA12 are not sufficient to implement both designs in one device. As a combination of a Slave Transmitter and a Slave Receiver with a high number of inputs and outputs is seldom requested, this should not be a problem.

The first PLC42VA12 design is an I²C-bus Slave Transmitter controller. With one or multiple 74HC165 devices, it forms an I²C-bus n-bit Input Expander. At one side the controller fully controls the I²C-Bus Slave Transmitter function, and at the other side it generates the control signals for the 74HC165. The 74HC165 devices can be cascaded to increase the number of inputs. Chapter. I²C-Bus Slave Transmitter Function gives a detailed description of the design.

The second design is an I²C-bus Slave Receiver controller. With one or multiple 74HC595 devices it forms an I²C-bus n-bit Output Expander. At one side the controller fully controls the I²C-Bus Slave Receiver function, and at the other side it generates the control signals for the 74HC595. The 74HC595 devices can be cascaded to increase the number of outputs. The section entitled I²C-Bus Slave Receiver Function describes in detail the design.

For design verification purposes, a bread-board has been made. This board contains all the devices to build a Slave Transmitter with 32 inputs and a Slave Receiver with 32 outputs. The main devices of the board are:

- A PLC42VA12 with the Slave Transmitter controller function,
- 4 PC74HC165 devices,
- 4 octal DIP-switches,
- A PLC42VA12 containing the Slave Receiver controller function,
- 4 PC72HC595 devices,
- 32 LED's.

CHARACTERISTICS OF THE I²C-BUS

The I²C-bus is a 2-way, 2-line communication between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). Both lines must be connected to a positive supply via a pull-up resistor when connected to the output stages of a device. Data transfer may be initiated only when the bus is not busy.

Bit Transfer

One data bit is transferred during each clock pulse. The data on the SDA line must remain stable during the HIGH period of the clock pulse as changes in the data line at this time will be interpreted as control signals (Figure 1, Bit Transfer). The maximum clock frequency is 100 kHz.

Start and Stop Conditions

Both data and clock lines remain HIGH when the bus is not busy. A HIGH-to-LOW transition of the data line, while the clock is HIGH is defined as the start condition (S). A LOW-to-HIGH transition of the data line while the clock is HIGH is defined as the stop condition (P). Figure 2, Definition of Start and Stop Conditions, gives the timing diagram.

System Configuration

A device generating a message is a transmitter, a device receiving a message is the receiver. The device that controls the message is the master and the devices which are controlled by the master are slaves. Figure 3, System Configuration, gives a block diagram of the system configuration.

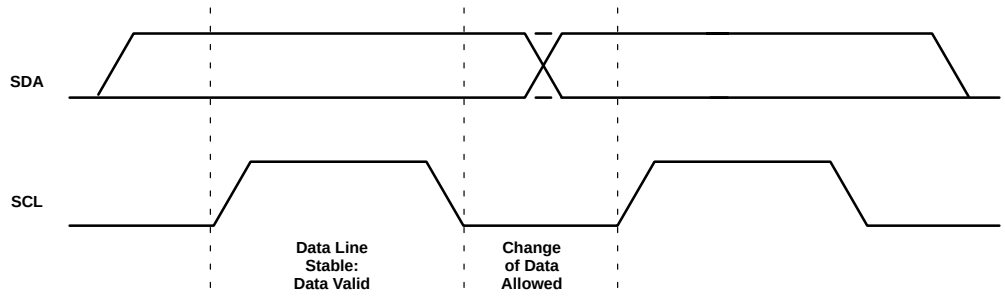


Figure 1. Bit Transfer

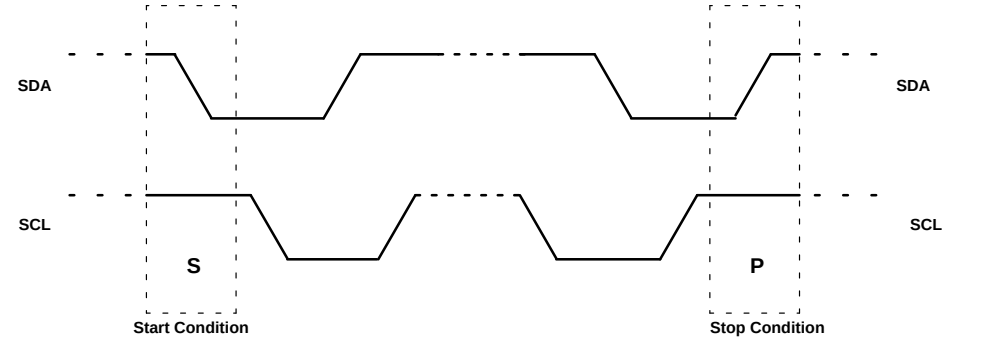


Figure 2. Definition of Start and Stop Conditions

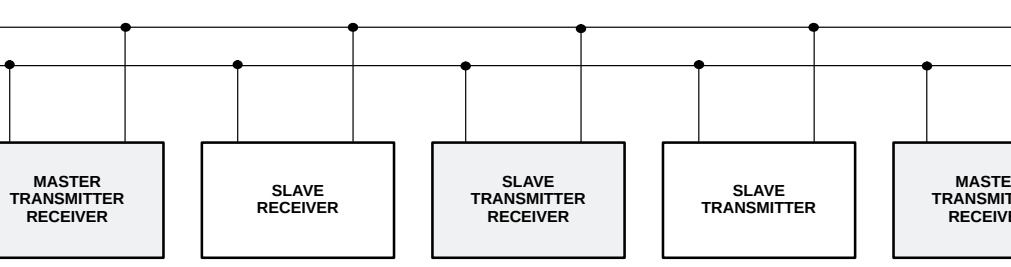


Figure 3. System Configuration

Acknowledge

The number of data bytes transferred between the start and the stop conditions from transmitter to receiver is not limited. Each byte of eight bits is followed by one acknowledge bit. The acknowledge bit is a HIGH level put on the bus by the transmitter whereas the master generates an extra acknowledge related clock pulse. A slave receiver which is addressed must generate an acknowledge after the reception of each byte. Also a master must generate an acknowledge after the reception of each byte that has been clocked out of the slave transmitter. The device that acknowledges has to pull down the SDA line during the acknowledge clock pulse, so that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Setup and hold times must be taken into account. A master receiver must signal an end of data to the transmitter by not generating an acknowledge on the last byte that has been clocked out of the slave. In this event the transmitter must leave the data line HIGH to enable the master to generate a stop condition (see Figure 4, Acknowledgement on the I²C-bus).

Formats

Data transfers follow the format shown in Figure 5 Data formats of the I²C-bus. After the START condition, the master sends the slave address. This address is 7 bits long, the eighth bit is a data direction bit (R/WN). A zero indicates a transmission (WRITE) and a one indicates a request for data (READ). A master always terminates a data transfer by a STOP condition. However, if a master still wishes to communicate on the bus, it can generate an other START condition and address an other slave without first generating a STOP condition. Various combinations of read and write formats are then possible within such a transfer.

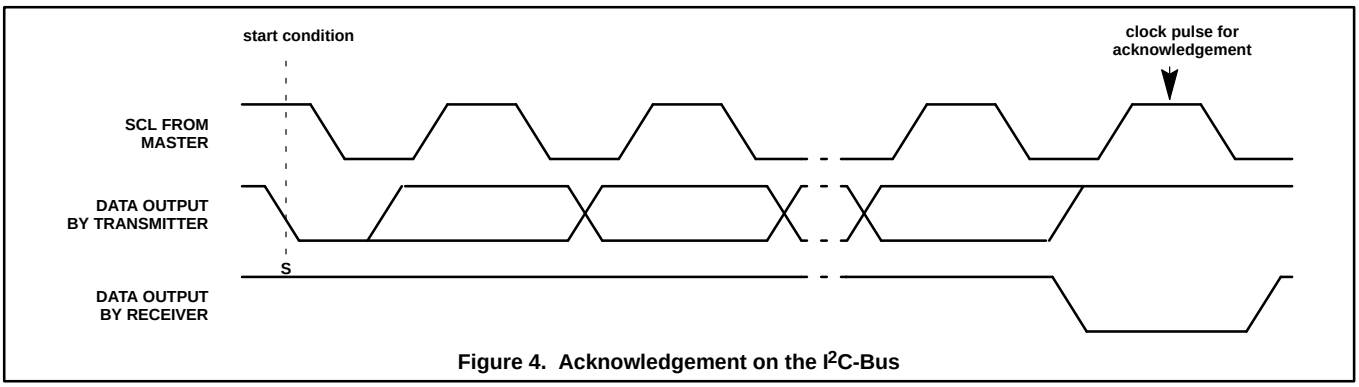
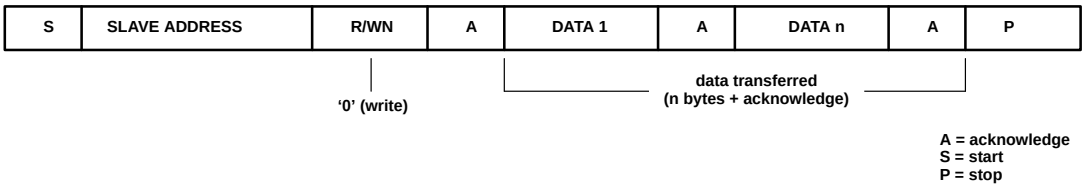


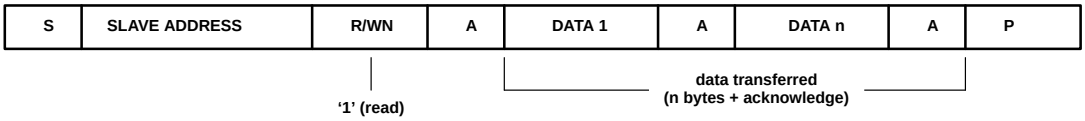
Figure 4. Acknowledgement on the I²C-Bus

Possible data transfer formats are:

a. Master transmitter to slave receiver. Direction is not changed.

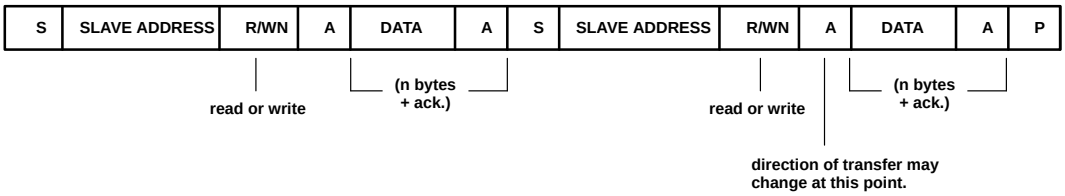


b. Master reads slave immediately after first byte.



At the moment of the first acknowledge, the master transmitter becomes a master receiver and the slave receiver becomes a slave transmitter. This acknowledge is still generated by the slave.

The STOP condition is generated by the master.



During a change of direction within a transfer, the START condition and the slave address are both repeated, but the R/W bit reversed. Start, stop, slave addresses and R/W bits are generated by the master.

Figure 5. Data formats of the I²C-Bus

COMMON BASIC FUNCTIONS

This section gives a number of common basic functions used in the designs. The report gives for each function the basic diagram, the SNAP description and the timing diagram if applicable. The following basic circuits are described:

- Oscillator
- SCL Edge detection
- Start/Stop detection

Oscillator

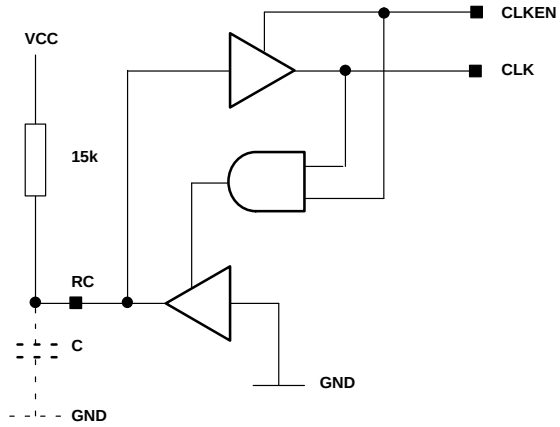
The design has two clock options, the internal oscillator and an external clock. For both options, the clock input CLK is used. A HIGH CLKEN input selects the internal oscillator and a LOW input the external clock. Without capacitor, we get the maximum frequency of the internal clock of 8 MHz. This frequency can be lowered by using a small capacitor C. Figure 6 Oscillator shows the diagram and the EQN file description of the oscillator.

SCL Edge Detection

The frequency of the system clock is much higher than the I²C-bus clock (SCL). This means, that most of the time the state machine is waiting for the edges of the SCL clock. This section describes the circuit that detects the HIGH and the LOW going edge of the SCL clock. The state machines synchronizes on the output pulses SCLH and SCLL.The detection network uses only two flip flops and two AND gates. Figure 7 SCL Timing Diagram SCL Edge Detection. gives the timing diagram of the edge detector and Figure 8 SCL Edge Detection. the diagram and the description of the EQN file.

Start/Stop Detection

A HIGH-to-LOW transition of the data line, while the clock is HIGH has been defined as the start condition (S) of the I²C-bus. A LOW-to-HIGH transition of the data line while the clock is HIGH has been defined as the stop condition (P). The easiest way to detect this start and stop condition is using asynchronous logic. The PLC42VA12 is very suited for this kind of solutions. Figure 2, Definition of Start and Stop Conditions, gives the timing diagram of these conditions.




```
@PINLIST
RC              B;
CLK             B;
CLKEN           I;

@LOGIC EQUATIONS
CLK.OW          =CLKEN
RC.OE           =CLK*CLKEN;
RC              =GND;
CLK             =RC;
```

Figure 6. Oscillator

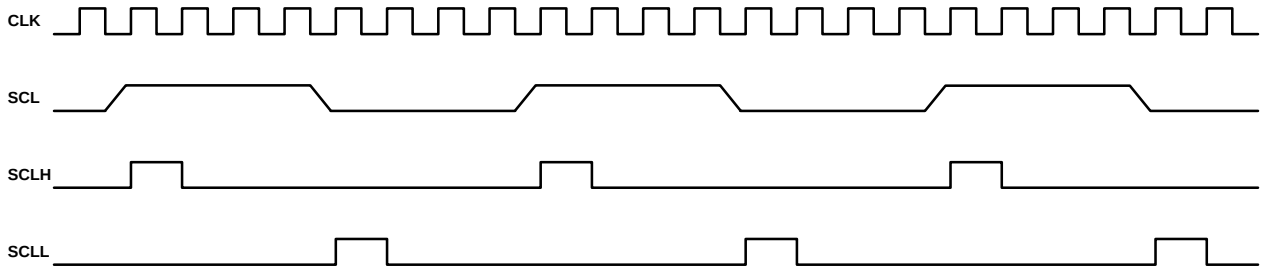
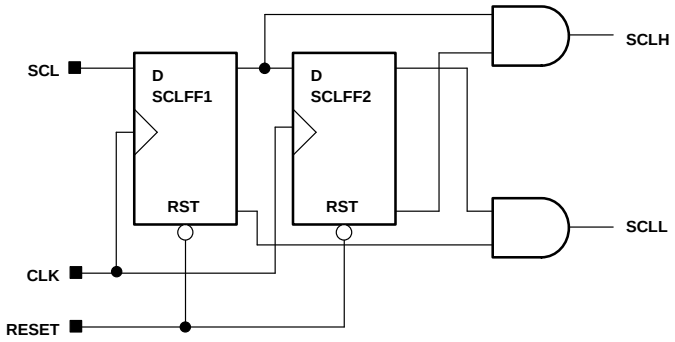


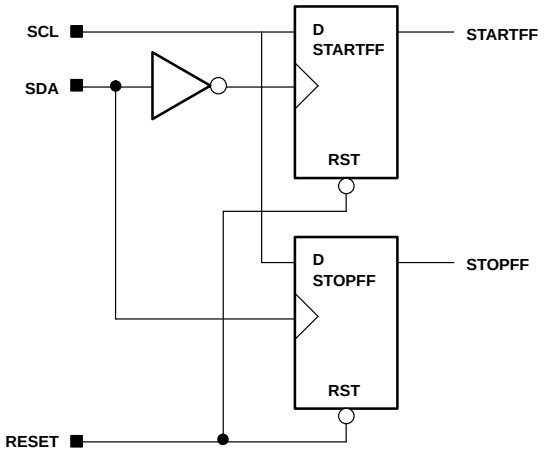
Figure 7. SCL Timing Diagram SCL Edge Detection




```
@PINLIST
SCL             I;
CLK             B;
RESET           I;

@LOGIC EQUATIONS
SCLFF1.D        =SCL;
SCLFF1.RST      =RESET;
SCLFF2.D        =SCLFF1;
SCLFF2.RST      =RESET;
SCLH            =SCLFF1*/SCLFF2;
SCLL            =/SCLFF1*SCLFF2;
```

Figure 8. SCL Edge Detection




```
@PINLIST
SCL             I;
SDA             B;
RESET           I;

@LOGIC EQUATIONS
STARTFF.CLK     =/SDA;
STARTFF.D       =SCL;
STARTFF.RST     =RESET;
STOPFF.CLK      =SDA;
STOPFF.D        =SCL;
STOPFF.RES      =RESET;
```

Figure 9. Start/Stop Detection

I²C-BUS SLAVE TRANSMITTER FUNCTION

The Slave Transmitter design provides remote input expansion for our Philips microcontroller families via the two-line serial bidirectional I²C-Bus. The I²C-Bus slave address is equal to the address of the PCF8574 (remote 8-bit I/O expander). The design will only acknowledge the read-mode.

The design handles the full slave read mode of the I²C-Bus and will generate the control signals for the 74HC165, an 8-bit parallel-in / serial-out shiftregister. This device is used to read the parallel input data and convert this into serial data. This data is written to the I²C-Bus. The total number of 74HC165 devices is almost unlimited.

With the three address selection inputs, the slave transmitter can be combined with multiple PCF8574 devices.

The design has a build-in clock oscillator. The section entitled Oscillator describes this circuit. If an external clock is required, the internal oscillator can be inhibited with the CLKEN-input.

Figure 17, Slave Transmitter EQN File, Figure 18, Slave Transmitter SCL File, and Figure 19, Slave Transmitter PIN File, give the design files. The sections SDA Control Slave Transmitter, I²C-Bus protocol Slave Transmitter and Interface with 74HC165 give a detailed description of parts of the design.

SDA Control Slave Transmitter

The SDA data line of the I²C-bus is a bidirectional line with a passive pull-up. This asks for a bidirectional open collector I/O line. As the PLC42VA12 has only 3-State I/Os, we need one of the advantages of the PLC42VA12 to get an open collector output. IN a PLC42VA12 each OE-input can be used as a logic input. With a LOW level (ground) at the input, the output has a LOW level when OE is true and is floating when OE is false. These are the characteristics of an open collector output.

Only one AND-gate controls the OE input of a bidirectional I/O. The design asks for multi-level logic. Figure 10 SDA Control Slave Transmitter shows how this input can be controlled by multi-level logic. The inputs ACKNOW, DATIN and SDAIN are auxiliary outputs of the PLC42VA12, used as inputs of the SDA control.

I²C-Bus protocol Slave Transmitter

The section entitled Formats describes the general data formats of the I²C-Bus protocol. Figure 11 I²C-Bus Protocol Slave Transmitter gives the protocol for this design. After a start condition, the master sends the slave address of the device. This 7 bits address consists of a fixed part and a programmable part. The first four bits are fixed (0100) and the three least significant bits are programmable. Three hardware address pins determine the final slave address. Up to 8 devices (or PCF8574) may be addressed by the master. After the slave address and a HIGH R/WN bit, the slave generates an acknowledge. At the next LOW SCL, the slave may start sending the first data byte. This byte will be acknowledged by the master. Also the next bytes will be acknowledged by the master. As after the acknowledge pulse the slave controls the SDA-line, the master can not generate a stop condition. The only way for a master to terminate a transmission, is not to acknowledge the last byte n. Then, the slave transmitter will release the SDA-line and the master can generate a stop condition.

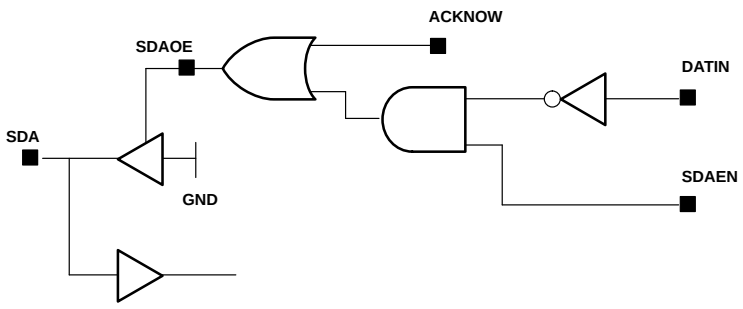
Interface with 74HC165

The 74HC/HCT165 is an 8-bit parallel load or serial-in shift register with complementary serial outputs (Q7 and Q7N) available from the last stage. When the parallel load (PLN) input is LOW, parallel data from the D0 to D7 inputs are loaded into the register asynchronously. When PLN is HIGH, data enters the register serially at the DS input and shifts one place to the right with each positive-going clock transition. This feature allows parallel-to-serial converter expansion by tying the Q7 output to the DS input of the succeeding stage.

The CLOUT output of the PLC42VA12 controls the clock of the 74HC165 and the PLOADN output controls the PLN input. The Q7 output of the 74HC165 is the data input DATIN of the PLC42VA12.

With this setup, the most significant bit of the data is the first bit that will be sent from the slave to the master. Figure 11 I²C-Bus Protocol Slave Transmitter gives the timing diagram of this interface.

When the slave address and the read bit have been detected, the controller generates the parallel load pulse PLOADN. After sending the firs bit (most significant bit of the transmission) it generates the first shift pulse CLOCKOUT. At the end of the first byte, the master generates an acknowledge. The second byte starts with a shift pulse CLOCKOUT. At each next LOW SCL level, this pulse is repeated. If at the end of the byte the master sends an acknowledge, then the next byte will be sent. A not acknowledge stops the procedure.




```
@PINLIST
SDA             B;
ACKNOW          I;
DATIN           I;
SDAEN           O;
SDAOE           O;

@LOGIC EQUATIONS
SDA             =GND;
SDA.OE          =SDAOE;
SDAOE           =ACKNOW+
                (/DATIN*SDAEN)
```

Figure 10. SDA Control Slave Transmitter

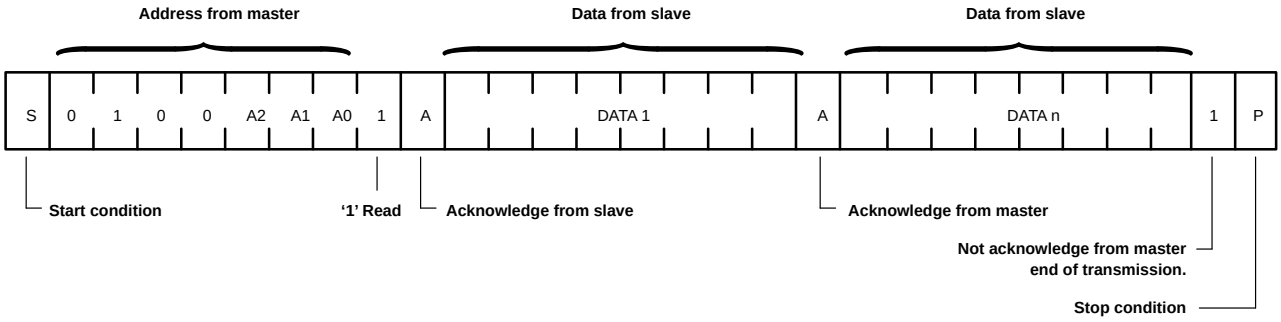


Figure 11. I²C-bus Protocol Slave Transmitter

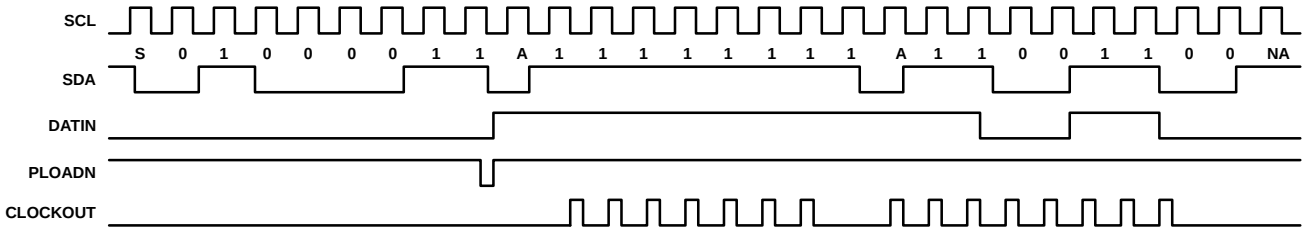


Figure 12. Interface with 74HC165

I²C-BUS SLAVE RECEIVER FUNCTION

The slave receiver design provides remote output expansion for our Philips micro controller families via the two-line serial bidirectional I²C-bus. The I²C-bus slave address is equal to the address of the PCF8574 (remote 8-bit I/O expander). The design will only acknowledge the write-mode.

The design handles the full slave write mode of the I²C-Bus and will generate the control signals for the 74HC595, an 8-bit serial-in / parallel-out shiftregister. This device is used to write the serial from the I²C-bus input to, and convert this into parallel data. The total number of 74HC595 devices is almost unlimited.

With the three address selection inputs, the slave receiver can be combined with multiple PCF8574 devices.

The design has a build-in clock oscillator. The section entitled Oscillator describes this circuit. If an external clock is required, the internal clock can be inhibited with the CLKEN-input. Figure 20, Slave Receiver EQN File, Figure 21, Slave Receiver SCL File, and Figure 22, Slave Receiver PIN File give the design files. The sections SDA Control Slave Receiver, Set-Reset Flip-Flop, I²C-Bus protocol Slave Receiver and Interface with 74HC595 give a detailed description of parts of the design.

SDA Control Slave Receiver

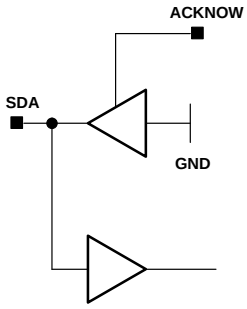
In the Slave Receiver protocol of the I²C-Bus, mainly the master controls the SDA-line. The Slave Receiver uses the SDA-line only for generating an acknowledge pulse. This is done after receiving its slave address with a write condition and all the following data bytes. Figure 13, SDA Control Slave Receiver gives the diagram and the description of the EQN-file.

Set-Reset Flip-Flop

The PLC42VA12 has 10 internal flip-flops. As the design needs an additional D-latch, this one has to be built out of gates. Figure 14, Set-Reset Flip-Flop Slave Receiver, gives the diagram and the equation file description of this function. In this example the signal STO7 is defined as an input, but in the final design this is a auxiliary output of the device.

I²C-Bus protocol Slave Receiver

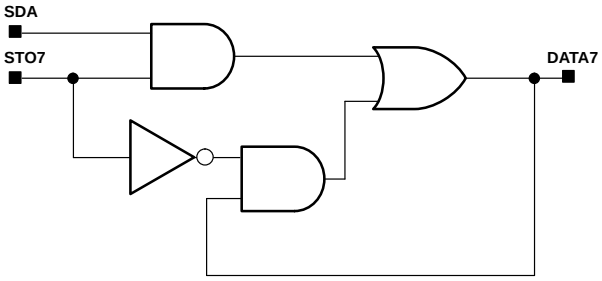
The section entitled Formats describes the general format of the I²C-Bus protocol. Figure 14, Set-Reset Flip-Flop Slave Receiver, gives the protocol for this design. After a start condition, the master sends the slave address of the device. This 7 bits address consists of a fixed part and a programmable part. The first four bits are fixed (0100) and the three least significant bits are programmable. Three hardware address pins determine the final slave address. Up to 8 devices (or PCF8574) may be addressed by the master. After the slave address and a LOW R/WN bit, the slave generates an acknowledge. At the next LOW SCL, the master starts sending the first data byte. This byte will be acknowledged by the slave. Also the next bytes will be acknowledged by the slave. The master terminates a transmission, by sending a stop condition or a restart condition.




```
@PINLIST
SDA             B;
ACKNOW          I;

@LOGIC EQUATIONS
SDA             =GND;
SDA.OE          =ACKNOW
```

Figure 13. SDA Control Slave Receiver




```
@PINLIST
SDA             B;
ST07            I;
DATA7           O;

@LOGIC EQUATIONS
DATA7 = ((STO7*SDA)+
        (DATA7*/ST07))
```

Figure 14. Set-Reset Flip-Flop Slave Receiver

Interface with 74HC595

The 74HC/HCT595 is an 8-stage serial shift register with a storage register and 3-State outputs. The shift register and storage register have separate clocks.

Data is shifted on the positive-going transitions of the SHCP input. The data in each register transfers to the storage register on a positive going transition of the STCP input. The shift register has a serial input (DS) and a serial standard output (Q7) for cascading. All 8 shift registers have an asynchronous reset (active LOW). The storage register has 8 parallel 3-State bus driver outputs. Data in the storage registers at the output whenever the output enable input (OEN) is LOW.

Four outputs of the PLC42VA12 control the inputs of the 74HC595. The RESOUT output the MRN input, CLKSTO the STCP input, CLKSHFT the SHCP and the DATOUT the DS input.

The I²C-Bus sends first the most significant bit of the transmission. Figure 16 I²C-Bus Interface with 74HC595 gives the timing diagram of this interface.

To set all outputs of the 74HC595 to a defined level, after power-on, the controller generates first a reset pulse at the RESOUT output and then a clock pulse CLKSTO for the storage register.

After a the slave address and the write bit (LOW) have been detected, during the next HIGH period of the SCL line there are three options. At the SDA line there can be the most significant bit of new data, the master generates a restart condition or the master generates a stop condition. This is also the case after each acknowledge.

This implies, that this first data bit must be stored. At the next LOW period of the SCL line we know whether we had data or restart/stop condition. If it was data, then we have to put this data at the DATOUT output and generate a clock pulse at the CLKSHFT output. At the next 7 HIGH periods of the SCL-line, data is valid and the controller generates a clock at the CLKSHFT output. At the end of the transmission, the master generates a stop condition or a restart. Then the stored data will be transferred to the storage register by a clock pulse at the CLKSTO output.

BREAD-BOARD I²C-BUS I/O EXPANDER

For design verification purposes, a bread-board has been designed. The board contains all the devices to build an I²C-Bus Slave Transmitter with 32 inputs and an I²C-Bus Slave Receiver with 32 outputs. The inputs can be set HIGH or LOW by 4 octal DIP-switches. The outputs are examined by 32 LED's. Figure 23, Schematic Diagram Bread-Board, gives the complete diagram of the bread-board.

The board has been designed for design verification only.

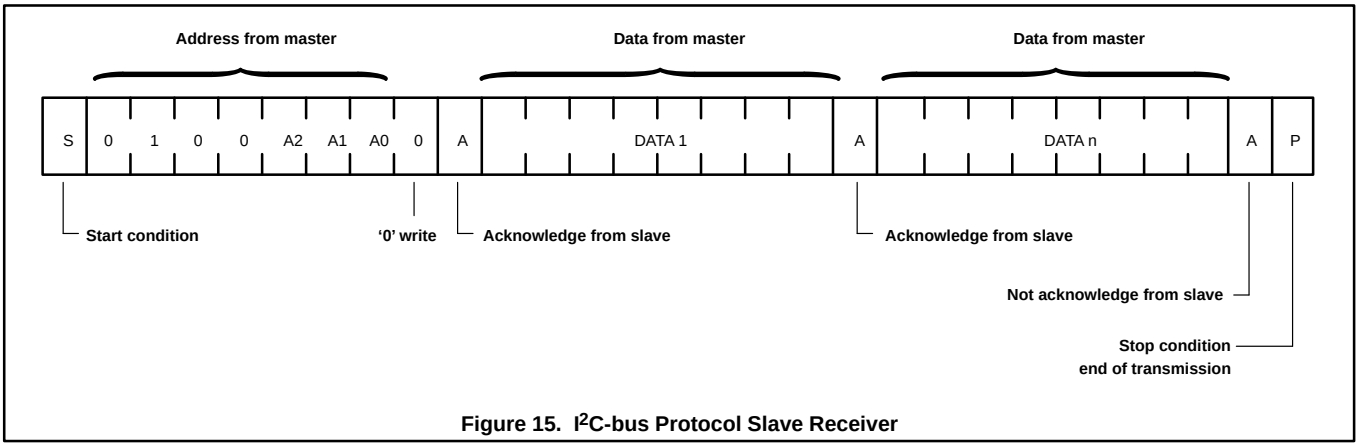


Figure 15. I²C-bus Protocol Slave Receiver

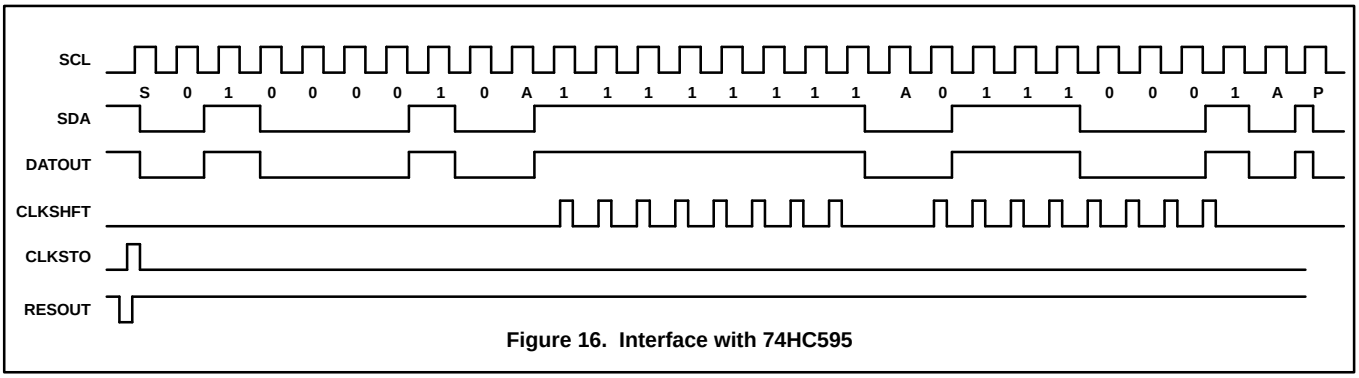


Figure 16. Interface with 74HC595

```
"
************************************************
*                                              *
*               Equation Entry File            *
*                                              *
* Project     : IIC                            *
* Function    : IIC-bus Slave Transmitter      *
*                                              *
* File Name   : IICTRANS.EQN                   *
* Design file : IICTRANS.SCL                   *
* Pin File    : IICTRANS.PIN                   *
*                                              *
* Date        : March 1993                     *
* Designer    : Aloys Schatorj                 *
* Company     : Philips Semiconductors         *
* Department  : PCALE                          *
* Place       : Eindhoven                      *
* Country     : The Netherlands                *
*                                              *
************************************************
"
@PINLIST
CLK       B;      "System clock"
RC        B;      "RC input internal clock"
CLKEN     I;      "Clock selection input"
SCL       I;      "IIC-BUS clock"
SDA       B;      "IIC-BUS data"
RESET     I;      "System reset"
ADD0      I;      "Address selection line"
ADD1      I;      "Address selection line"
ADD2      I;      "Address selection line"
DATIN     I;      "Data from input shift-register"
CLOCKOUT  O;      "Clock to input shift-register"
PLOADN    O;      "Parallel load to input shift-register"
SDAOE     O;      "Enable line SDA I/O"
SDAEN     O;      "Enable condition SDA caused by data"
ACKNOW    O;      "Acknowledge data"
STATEST   O;      "Reset start and stop FF"

@GROUPS
@TRUTHTABLE
@LOGIC EQUATIONS
CLK.OE      =  CLKEN   ;
RC.OE       =  CLK*CLKEN;
RC          =  GND     ;
CLK         =  RC      ;
STARTFF.CLK = /SDA     ;
STARTFF.D   =  SCL     ;
STARTFF.RST =  STATEST ;
SCLFF1.D    =  SCL     ;
SCLFF1.RST  =  RESET   ;
SCLFF2.D    =  SCLFF1  ;
SCLFF2.RST  =  RESET   ;
SCLH        = SCLFF1*/SCLFF2 ;
SCLL        = /SCLFF1*SCLFF2 ;
Q0.RST      =  RESET   ;
Q1.RST      =  RESET   ;
Q2.RST      =  RESET   ;
Q3.RST      =  RESET   ;
Q4.RST      =  RESET   ;
Q5.RST      =  RESET   ;
SDA         =  GND     ;
SDA.OE      =  SDAOE   ;
PLOADN      = /PLOAD   ;
SDAOE       =  ACKNOW + (/DATIN*SDAEN)   ;
```

Figure 17. Equation Entry File Slave Transmitter (1 of 3)

```
@INPUT VECTORS

@OUTPUT VECTORS
         [ACKNOW, CLOCKOUT, PLOAD, SDAEN]
ACK     =   1       –        –       –   B;
CLKOUT  =   –       1        –       1   B;
SDAENA  =   –       –        –       1   B;
PLOUT   =   –       –        1       1   B;

@STATE VECTORS
        [Q5, Q4, Q3, Q2, Q1, Q0] JKFFR
INITL     =  00 H;
INIT      =  3F H;
WAIT      =  3E H;
WAIT1     =  3D H;
ADDBIT6   =  01 H;
ADDBIT5   =  02 H;
ADDBIT4   =  03 H;
ADDBIT3   =  04 H;
ADDBIT2   =  05 H;
ADDBIT1   =  06 H;
ADDBIT0   =  07 H;
RWBIT     =  08 H;
READMOD   =  09 H;
ACKBITR   =  0A H;
READ7L    =  0B H;
READ7C    =  10 H;
READ7     =  11 H;
READ6C    =  12 H;
READ6     =  13 H;
READ5C    =  14 H;
READ5     =  15 H;
READ4C    =  16 H;
READ4     =  17 H;
READ3C    =  18 H;
READ3     =  19 H;
READ2C    =  1A H;
READ2     =  1B H;
READ1C    =  1C H;
READ1     =  1D H;
READ0C    =  1E H;
READ0     =  1F H;
ACKPLS    =  20 H;
ACKPLSW   =  21 H;

@TRANSITIONS
WHILE   [INITL]
        IF    []           THEN  [INIT]
WHILE   [INIT]
        IF    []           THEN  [WAIT]
WHILE   [WAIT]                                  WITH  [STATEST]
        IF    [STARTFF]    THEN  [WAIT1]
WHILE   [WAIT1]                                 WITH  [STATEST]
        IF    []           THEN  [ADDBIT6]
WHILE   [ADDBIT6]
        IF    [SCLH*SDA]   THEN  [WAIT]
        IF    [SCLH*/SDA]  THEN  [ADDBIT5]
WHILE   [ADDBIT5]
        IF    [SCLH*/SDA]  THEN  [WAIT]
        IF    [SCLH*SDA]   THEN  [ADDBIT4]
WHILE   [ADDBIT4]
        IF    [SCLH*SDA]   THEN  [WAIT]
        IF    [SCLH*/SDA]  THEN  [ADDBIT3]
WHILE   [ADDBIT3]
        IF    [SCLH*SDA]   THEN  [WAIT]
        IF    [SCLH*/SDA]  THEN  [ADDBIT2]
WHILE   [ADDBIT2]
        IF    [SCLH*/((SDA*ADD2)+(/SDA*/ADD2))] THEN  [WAIT]
        IF    [SCLH*((SDA*ADD2)+(/SDA*/ADD2))]  THEN  [ADDBIT1]
```

Figure 17. Equation Entry File Slave Transmitter (2 of 3)

```
WHILE   [ADDBIT1]
        IF    [SCLH*/((SDA*ADD1)+(/SDA*/ADD1))] THEN  [WAIT]
        IF    [SCLH*((SDA*ADD1)+(/SDA*/ADD1))]  THEN  [ADDBIT0]
WHILE   [ADDBIT0]
        IF    [SCLH*/((SDA*ADD0)+(/SDA*/ADD0))] THEN  [WAIT]
        IF    [SCLH*((SDA*ADD0)+(/SDA*/ADD0))]  THEN  [RWBIT]
WHILE   [RWBIT]
        IF    [SCLH*SDA]   THEN  [READMOD]
        IF    [SCLH*/SDA]  THEN  [WAIT]
WHILE   [READMOD]
        IF    [SCLL]       THEN  [ACKBITR]
WHILE   [ACKBITR]                                  WITH  [ACK]
        IF    [SCLL]       THEN  [READ7L]
WHILE   [READ7L]                                   WITH  [PLOUT]
        IF    []           THEN  [READ7]
WHILE   [READ7C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ7]
WHILE   [READ7]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ6C]
WHILE   [READ6C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ6]
WHILE   [READ6]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ5C]
WHILE   [READ5C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ5]
WHILE   [READ5]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ4C]
WHILE   [READ4C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ4]
WHILE   [READ4]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ3C]
WHILE   [READ3C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ3]
WHILE   [READ3]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ2C]
WHILE   [READ2C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ2]
WHILE   [READ2]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ1C]
WHILE   [READ1C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ1]
WHILE   [READ1]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [READ0C]
WHILE   [READ0C]                                   WITH  [CLKOUT]
        IF    []           THEN  [READ0]
WHILE   [READ0]                                    WITH  [SDAENA]
        IF    [SCLL]       THEN  [ACKPLS]
WHILE   [ACKPLS]
        IF    [SCLH*/SDA]  THEN  [ACKPLSW]
        IF    [SCLH*SDA]   THEN  [WAIT]
WHILE   [ACKPLSW]
        IF    [SCLL]       THEN  [READ7C]
```

Figure 17. Equation entry file slave transmitter (3 of 3)

```
************************************************
*                                              *
*       Simulation Control Language File       *
*                                              *
* Project     : IIC                            *
* Function    : IIC-bus Slave Transmitter      *
*                                              *
* File Name   : IICTRANS.SCL                   *
* Design file : IICTRANS.EQN                   *
* Pin File    : IICTRANS.PIN                   *
*                                              *
* Date        : March 1993                     *
* Designer    : Aloys Schatorj                 *
* Company     : Philips Semiconductors         *
* Department  : PCALE                          *
* Place       : Eindhoven                      *
* Country     : The Netherlands                *
*                                              *
************************************************
*
 P CLK, RESET, SCL, SDA, DATIN, SDAEN, CLOCKOUT, PLOADN, ACKNOW,
#  STARTFF, STATEST, Q0, Q1, Q2, Q3, Q4,
#  RC, CLKEN, ADD0, ADD1, ADD2, VCC
*  SCLH, SCLL,
 PCO
*** Initialisation ***
 BUSI SDA
 BUSI CLK
 BUSO RC
 S  0 (50, 100, ETC)CLK
 S  0 (75)RESET
 S  0 (500, 1000, ETC)SCL
 ST 1 (DATIN)
 ST 1 (VCC)
 ST 1 (SDA)
 ST 0 (CLKEN)
 ST 001  (ADD2, ADD1, ADD0)
 SU TIME = 1225
*** Generate start condition ***
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500
*** receive device address with read (01000011). ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+4000
 ST 1 (SDA)
 SU TIME = *+2000
*** Generate acknowledge from slave ***
 BUSO SDA
 SU TIME = *+1000
*** Transmit 8 bits data word 11111111 ***
 ST 1 (DATIN)
 SU TIME = *+8000
*** Generate acknowledge from master ***
 BUSI SDA
 ST 0 (SDA)
 SU TIME = *+1000
```

Figure 18. .SCL File Slave Transmitter (1 of 2)

```
*** Transmit 8 bits data word 11001100 ***
 BUSO SDA
 ST 1 (DATIN)
 SU TIME = *+2000
 ST 0 (DATIN)
 SU TIME = *+2000
 ST 1 (DATIN)
 SU TIME = *+2000
 ST 0 (DATIN)
 SU TIME = *+2000

*** Generate no acknowledge from master (end of transmition) ***
 BUSI SDA
 ST 1 (SDA)
 SU TIME = *+1000

*** Wait for new start condition ***
 SU TIME = *+3000

*** Generate new start condition ***
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500

*** receive device address with write (01000010). ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+4000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+1000

*** Wait for new start condition ***
 SU TIME = *+3000

*** Generate new start condition ***
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500

*** Receive wrong device address (0110000). ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+2000
 ST 0 (SDA)
 SU TIME = *+6000

*** Test internal clock ***
 BUSO CLK, RC
 ST 1 (CLK)
 ST 1 (CLKEN)
 SU TIME = *+ 1000
 F
```

Figure 18. .SCL File Slave Transmitter (2 of 2)

```
************************************************
*                                              *
*       Pinning File                           *
*                                              *
* Project     : IIC                            *
* Function    : IIC-bus Slave Transmitter      *
*                                              *
* File Name   : IICTRANS.EQN                   *
* Design file : IICTRANS.SCL                   *
* Pin File    : IICTRANS.PIN                   *
*                                              *
* Date        : March 1993                     *
* Designer    : Aloys Schatorje                *
* Company     : Philips Semiconductors         *
* Department  : PCALE                          *
* Place       : Eindhoven                      *
* Country     : The Netherlands                *
*                                              *
************************************************
"

Device      =C42VA12
Pin2        =RESET
Pin3        =ADD0
Pin4        =ADD1
Pin5        =ADD2
Pin8        =CLKEN
Pin9        =SCL
Pin10       =SDA
Pin11       =CLOCKOUT
Pin14       =CLK
Pin15       =RC
Pin18       =DATIN
Pin19       =ACKNOW
Pin20       =SDAOE
Pin21       =SDAEN
Pin22       =STATEST
Pin23       =PLOADN
```

Figure 19. .PIN File Slave Transmitter

```
"
************************************************
*                                              *
*               Equation Entry File            *
*                                              *
* Project     : IIC                            *
* Function    : IIC-bus Slave Receiver         *
*                                              *
* File Name   : IICRECEI.EQN                   *
* Design file : IICRECEI.SCL                   *
* Pin File    : IICRECEI.PIN                   *
*                                              *
* Date        : March 1993                     *
* Designer    : Aloys Schatorj                 *
* Company     : Philips Semiconductors         *
* Department  : PCALE                          *
* Place       : Eindhoven                      *
* Country     : The Netherlands                *
*                                              *
************************************************
"
@PINLIST
CLK       B;      "System clock"
RC        B;      "RC input internal clock"
CLKEN     I;      "Clock selection input"
SCL       I;      "IIC-BUS clock"
SDA       B;      "IIC-BUS data"
RESET     I;      "System reset"
ADD0      I;      "Address selection line"
ADD1      I;      "Address selection line"
ADD2      I;      "Address selection line"
CLKSHFT   O;      "Clock to output shift-register"
CLKSTO    O;      "Parallel load into output register"
DATOUT    O;      "Data to output shift register"
DATA7     O;      "Output data RSFF bit7"
STO7      O;      "Store pulse DATA7 FF"
OUT7      O;      "Enable DATA7 FF data"
RESOUT    O;      "Reset output shift register"
ACKNOW    O;      "Acknowledge data"
STATEST   O;      "Reset start and stop FF"

@GROUPS
@TRUTHTABLE
@LOGIC EQUATIONS
CLK.OE      =  CLKEN   ;
RC.OE       =  CLK*CLKEN;
RC          =  GND     ;
CLK         =  RC      ;
STARTFF.CLK = /SDA     ;
STARTFF.D   =  SCL     ;
STARTFF.RST =  STATEST ;
STOPFF.CLK  =  SDA     ;
STOPFF.D    =  SCL     ;
STOPFF.RST  =  STATEST ;
SCLFF1.D    =  SCL     ;
SCLFF1.RST  =  RESET   ;
SCLFF2.D    =  SCLFF1  ;
SCLFF2.RST  =  RESET   ;
SCLH        = SCLFF1*/SCLFF2 ;
SCLL        = /SCLFF1*SCLFF2 ;
DATA7       =  ((STO7*SDA)+DATA7*/STO7) ;
Q0.RST      =  RESET   ;
Q1.RST      =  RESET   ;
Q2.RST      =  RESET   ;
Q3.RST      =  RESET   ;
Q4.RST      =  RESET   ;
Q5.RST      =  RESET   ;

SDA         =  GND     ;
SDA.OE      =  ACKNOW  ;
DATOUT      =  (SDA*/OUT7) + (DATA7*OUT7) ;
RESOUT      = /RESOUTN ;
```

Figure 20. Equation Entry File Slave Receiver (1 of 3)

```
@INPUT VECTORS

@OUTPUT VECTORS
         [CLKSHFT, CLKSTO, RESOUTN, STATEST, STO7, OUT7]
CLKSHIFT =  1        –       –        –       –      –   B;
CLKSTOR  =  –        1       –        1       –      –   B;
RESO     =  –        –       1        –       1      –   B;
STATST   =  –        –       –        1       –      –   B;
STASTO7  =  –        –       –        1       1      –   B;
OUTBIT7  =  –        –       –        –       –      1   B;
OUTCLK7  =  1        –       –        –       –      1   B;

@STATE VECTORS
        [Q5, Q4, Q3, Q2, Q1, Q0] JKFFR
INITL     =  00 H;
INIT      =  3F H;
INIT1     =  3E H;
INIT2     =  3D H;
WAIT      =  3C H;
WAIT1     =  3B H;
ADDBIT6   =  01 H;
ADDBIT5   =  02 H;
ADDBIT4   =  03 H;
ADDBIT3   =  04 H;
ADDBIT2   =  05 H;
ADDBIT1   =  06 H;
ADDBIT0   =  07 H;
RWBIT     =  08 H;
WRIDMOD   =  09 H;
ACKBITTR  =  0A H;
TESTSTA   =  0B H;
TESTSTA1  =  0C H;
TESTSTA2  =  0D H;
OUTB7     =  0E H;
CLKB7     =  0F H;
WRID6     =  10 H;
WRID6C    =  11 H;
WRID5     =  12 H;
WRID5C    =  13 H;
WRID4     =  14 H;
WRID4C    =  15 H;
WRID3     =  16 H;
WRID3C    =  17 H;
WRID2     =  18 H;
WRID2C    =  19 H;
WRID1     =  1A H;
WRID1C    =  1B H;
WRID0     =  1C H;
WRID0C    =  1D H;

@TRANSITIONS
WHILE   [INITL]
        IF    []           THEN  [INIT]
WHILE   [INIT]
        IF    []           THEN  [INIT1]
WHILE   [INIT1]                                 WITH  [RESO]
        IF    []           THEN  [INIT2]
WHILE   [INIT2]                                 WITH  [CLKSTOR]
        IF    []           THEN  [WAIT]
WHILE   [WAIT]                                  WITH  [STATST]
        IF    [STARTFF]    THEN  [WAIT1]
WHILE   [WAIT1]                                 WITH  [STATST]
        IF    []           THEN  [ADDBIT6]
WHILE   [ADDBIT6]
        IF    [SCLH*SDA]   THEN  [WAIT]
        IF    [SCLH*/SDA]  THEN  [ADDBIT5]
WHILE   [ADDBIT5]
        IF    [SCLH*/SDA]  THEN  [WAIT]
        IF    [SCLH*SDA]   THEN  [ADDBIT4]
```

Figure 20. Equation Entry File Slave Receiver (2 of 3)

```
WHILE   [ADDBIT4]
        IF    [SCLH*SDA]   THEN  [WAIT]
        IF    [SCLH*/SDA]  THEN  [ADDBIT3]
WHILE   [ADDBIT3]
        IF    [SCLH*SDA]   THEN  [WAIT]
        IF    [SCLH*/SDA]  THEN  [ADDBIT2]
WHILE   [ADDBIT2]
        IF    [SCLH*/((SDA*ADD2)+(/SDA*/ADD2))] THEN  [WAIT]
        IF    [SCLH*((SDA*ADD2)+(/SDA*/ADD2))]  THEN  [ADDBIT1]
WHILE   [ADDBIT1]
        IF    [SCLH*/((SDA*ADD1)+(/SDA*/ADD1))] THEN  [WAIT]
        IF    [SCLH*((SDA*ADD1)+(/SDA*/ADD1))]  THEN  [ADDBIT0]
WHILE   [ADDBIT0]
        IF    [SCLH*/((SDA*ADD0)+(/SDA*/ADD0))] THEN  [WAIT]
        IF    [SCLH*((SDA*ADD0)+(/SDA*/ADD0))]  THEN  [RWBIT]
WHILE   [RWBIT]
        IF    [SCLH*/SDA]  THEN  [WRIDMOD]
        IF    [SCLH*SDA]   THEN  [WAIT]
WHILE   [WRIDMOD]
        IF    [SCLL]       THEN  [ACKBITTR]
WHILE   [ACKBITTR]                                  WITH  [ACKNOW]
        IF    [SCLL]       THEN  [TESTSTA]
WHILE   [TESTSTA]                                   WITH  [STATST]
        IF    [SCLH]       THEN  [TESTSTA1]
WHILE   [TESTSTA1]                                  WITH  [STASTO7]
        IF    []           THEN  [TESTSTA2]
WHILE   [TESTSTA2]                                  WITH  [STATST]
        IF    [STARTFF]    THEN  [INIT2]
        IF    [STOPFF]     THEN  [INIT2]
        IF    [SCLL]       THEN  [OUTB7]
WHILE   [OUTB7]                                     WITH  [OUTBIT7]
        IF    []           THEN  [CLKB7]
WHILE   [CLKB7]                                     WITH  [OUTCLK7]
        IF    []           THEN  [WRID6]
WHILE   [WRID6]
        IF    [SCLH]       THEN  [WRID6C]
WHILE   [WRID6C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRID5]
WHILE   [WRID5]
        IF    [SCLH]       THEN  [WRID5C]
WHILE   [WRID5C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRID4]
WHILE   [WRID4]
        IF    [SCLH]       THEN  [WRID4C]
WHILE   [WRID4C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRID3]
WHILE   [WRID3]
        IF    [SCLH]       THEN  [WRID3C]
WHILE   [WRID3C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRID2]
WHILE   [WRID2]
        IF    [SCLH]       THEN  [WRID2C]
WHILE   [WRID2C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRID1]
WHILE   [WRID1]
        IF    [SCLH]       THEN  [WRID1C]
WHILE   [WRID1C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRID0]
WHILE   [WRID0]
        IF    [SCLH]       THEN  [WRID0C]
WHILE   [WRID0C]                                    WITH  [CLKSHIFT]
        IF    []           THEN  [WRIDMOD]
```

Figure 20. Equation Entry File Slave Receiver (3 of 3)

```
************************************************
*                                              *
*       Simulation Control Language File       *
*                                              *
* Project     : IIC                            *
* Function    : IIC-bus Slave Receiver         *
*                                              *
* File Name   : IICRECEI.SCL                   *
* Design file : IICRECEI.EQN                   *
* Pin File    : IICRECEI.PIN                   *
*                                              *
* Date        : March 1993                     *
* Designer    : Aloys Schatorj                 *
* Company     : Philips Semiconductors         *
* Department  : PCALE                          *
* Place       : Eindhoven                      *
* Country     : The Netherlands                *
*                                              *
************************************************
 P CLK, RESET, SCL, SDA, DATOUT, CLKSHFT, CLKSTO, RESOUT, ACKNOW,
#  STATEST, DATA7, OUT7, STO7, STARTFF, STOPFF, Q0, Q1, Q2, Q3, Q4, Q5,
#  RC, CLKEN, ADD0, ADD1, ADD2, VCC
*  SCLH, SCLL,
 PCO
*** Initialisation ***
 BUSI SDA
 BUSI CLK
 BUSO RC
 S  0 (50, 100, ETC)CLK
 S  0 (75)RESET
 S  0 (500, 1000, ETC)SCL
 ST 1 (VCC)
 ST 1 (SDA)
 ST 0 (CLKEN)
 ST 001  (ADD2, ADD1, ADD0)
 SU TIME = 1225

*** Generate start condition ***
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500

*** receive device address with write (01000010) ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+4000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+1000

*** Generate acknowledge from slave ***
 BUSO SDA
 SU TIME = *+1000

*** Receive 8 bits data word 11111111 ***
 BUSI SDA
 ST 1 (SDA)
 SU TIME = *+8000

*** Generate acknowledge from slave ***
 BUSO SDA
 SU TIME = *+1000
```

Figure 21. .SCL file slave receiver (1 of 3)

```
*** Receive 8 bits data word 11001100 ***
 BUSI SDA
 ST 1 (SDA)
 SU TIME = *+2000
 ST 0 (SDA)
 SU TIME = *+2000
 ST 1 (SDA)
 SU TIME = *+2000
 ST 0 (SDA)
 SU TIME = *+2000

*** Generate acknowledge from slave ***
 BUSO SDA
 SU TIME = *+1000

*** Receive 8 bits data word 01110001 ***
 BUSI SDA
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+3000
 ST 0 (SDA)
 SU TIME = *+3000
 ST 1 (SDA)
 SU TIME = *+1000

*** Generate acknowledge from slave ***
 BUSO SDA
 SU TIME = *+1000

*** Generate new start condition ***
 BUSI SDA
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500

*** receive device address with read (01000011) ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+4000
 ST 1 (SDA)
 SU TIME = *+2000

*** Wait for new start condition ***
 SU TIME = *+3000

*** Generate new start condition ***
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500

*** receive device address with write (01000010) ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+4000
 ST 1 (SDA)
 SU TIME = *+1000
 ST 0 (SDA)
 SU TIME = *+1000

*** Generate acknowledge from slave ***
 BUSO SDA
 SU TIME = *+1000
```

Figure 21. .SCL File Slave Receiver (2 of 3)

```
*** Generate stop condition ***
 ST 0 (SDA)
 SU TIME = *+500
 ST 1 (SDA)
 SU TIME = *+500
*** Wait for new start condition ***
 SU TIME = *+3000
*** Generate new start condition ***
 ST 1 (SDA)
 SU TIME = *+500
 ST 0 (SDA)
 SU TIME = *+500
*** receive wrong device address (0110000). ***
 ST 0 (SDA)
 SU TIME = *+1000
 ST 1 (SDA)
 SU TIME = *+2000
 ST 0 (SDA)
 SU TIME = *+6000
*** Test internal clock ***
 BUSO CLK, RC
 ST 1 (CLK)
 ST 1 (CLKEN)
 SU TIME = *+ 1000
 F
```

Figure 21. .SCL File Slave Receiver (3 of 3)

```
************************************************
*                                              *
*       Pinning File                           *
*                                              *
* Project     : IIC                            *
* Function    : IIC-bus Slave Receiver         *
*                                              *
* File Name   : IICRECEI.SCL                   *
* Design file : IICRECEI.EQN                   *
* Pin File    : IICRECEI.PIN                   *
*                                              *
* Date        : March 1993                     *
* Designer    : Aloys Schatorje                *
* Company     : Philips Semiconductors         *
* Department  : PCALE                          *
* Place       : Eindhoven                      *
* Country     : The Netherlands                *
*                                              *
************************************************
"

Device     =C42VA12
Pin2       =RESET
Pin3       =ADD0
Pin4       =ADD1
Pin5       =ADD2
Pin8       =CLKEN
Pin9       =SCL
Pin10      =SDA
Pin11      =CLKSHFT
Pin14      =CLK
Pin15      =RC
Pin16      =DATOUT
Pin17      =CLKSTO
Pin18      =RESOUT
Pin19      =ACKNOW
Pin20      =OUT7
Pin21      =DATA7
Pin22      =STATEST
Pin23      =STO7
```

Figure 22. .PIN File Slave Receiver

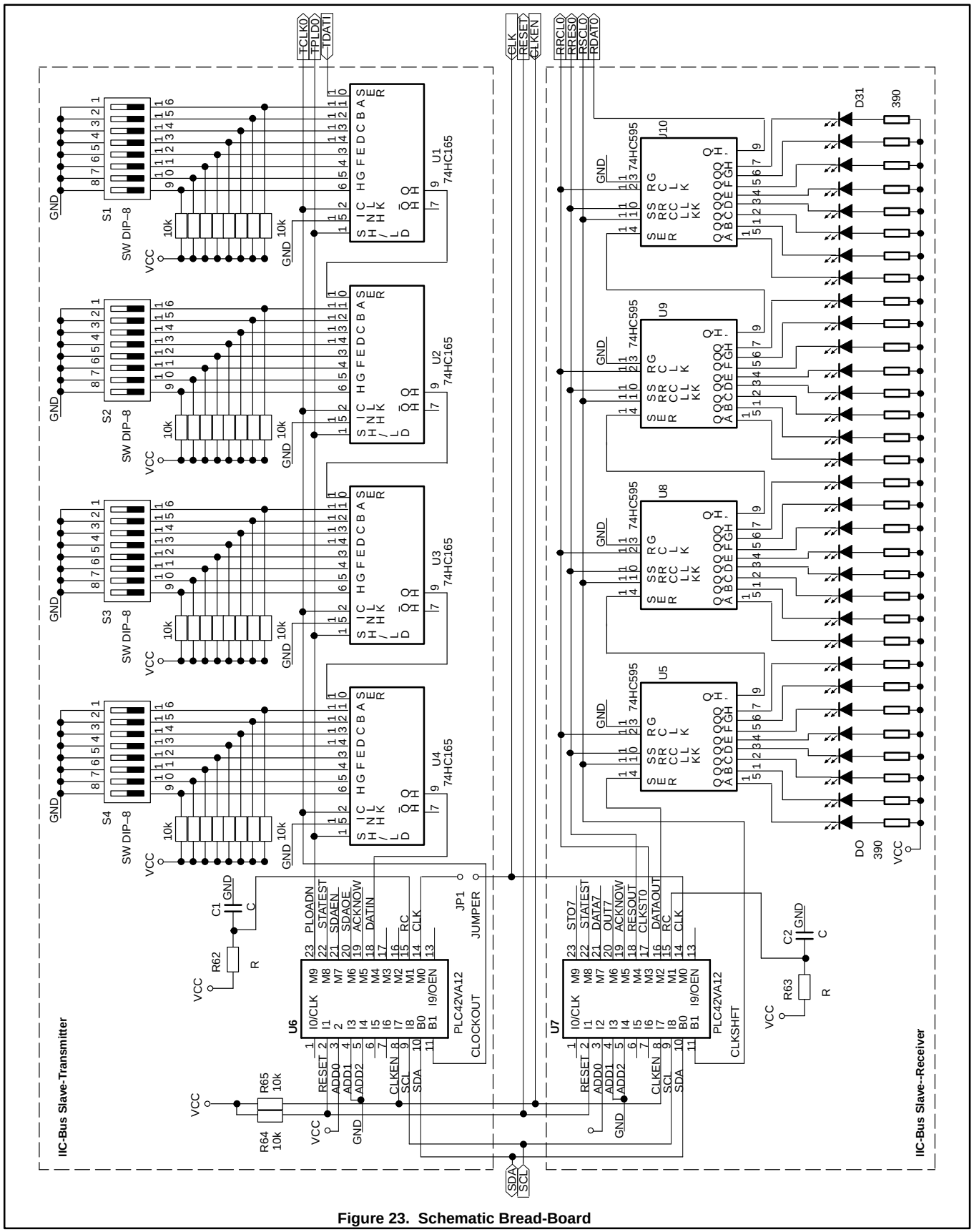


Figure 23. Schematic Bread-Board