I2C Core Design Doc

# Intent

The intent of this block is to handle all interaction with the I2C SDA and SCL lines. This will take

# Ports

|  |  |  |  |
| --- | --- | --- | --- |
| Port | Size | Dir | Function |
| PCLK | 1 | In | Peripheral clock, full Freq |
| RSTn | 1 | In | Reset = 0 |
| Ctrl\_i | 8 | In | Control register, used to select I2C events (Start, Stop, etc.) |
| Ctrl\_o | 8 | Out | Control register readback |
| Data\_i | 8 | In | Data input to be transmitted, must be stable for 1 clock cycle when I2C is started. |
| Data\_o | 8 | Out | Data output, when Receive byte is complete. This is the same internal register as Data\_i. |
| SDAI | 1 | In | SDA line input, Y from BiBuf |
| SDAO | 1 | Out | SDA line output, D to BiBuf |
| SDAE | 1 | Out | SDA output enable, E to BiBuf |
| SCLI | 1 | In | SCL line input, Y from BiBuf |
| SCLO | 1 | Out | SCL line output, D to BiBuf |
| SCLE | 1 | Out | SCL output enable, E to BiBuf |
| Int | 1 | Out | Interrupt when something needs doing (finish byte, ACK fail, etc.) |

# I2C Core - Base

## Internal Register

### Manual Control Registers

#### Control Register

|  |  |  |
| --- | --- | --- |
| Bit | R/W | Function |
| 7 | R | 0 : I2C bus is idle  1 : I2C bus is busy by either this core or some other I2C master |
| 6 - 5 | R | I2C Core Status  00 : Idle  01 : Busy doing action according to bits 4-2  10 : Interrupt active, waiting for next command, holding I2C bus  11 : Interrupt active with ACK failed, waiting for next command, holding I2C bus |
| 4 - 2 | W | 000 : NOP, No change to SDA or SCL but Interrupt will fire after 1 I2C clock  001 : Start, if I2C bus is idle, Start event will be sent.  010 : Stop, if status shows this I2C is in control, Stop signal will be sent.  011 : Repeated Start, if status shows this I2C is in control, RStart signal will be sent.  100 : Data will be written to the I2C bus, SDA Enable is enabled.  101 : Data will be read from the I2C bus, SDA Enable is disabled. |
| 1 | W | 0 : This I2C core is idle/listening  1 : Initiate I2C event with this I2C core, will be set to 0 upon completion |
| 0 | W | 0 : I2C core is disabled. Will neither transmit nor monitor I2C bus.  1 : I2C core is enabled.  \* Toggling this is like a soft reset. Probably. |

#### Address Register

10 bit

#### Data Register 0

#### Data Register 1

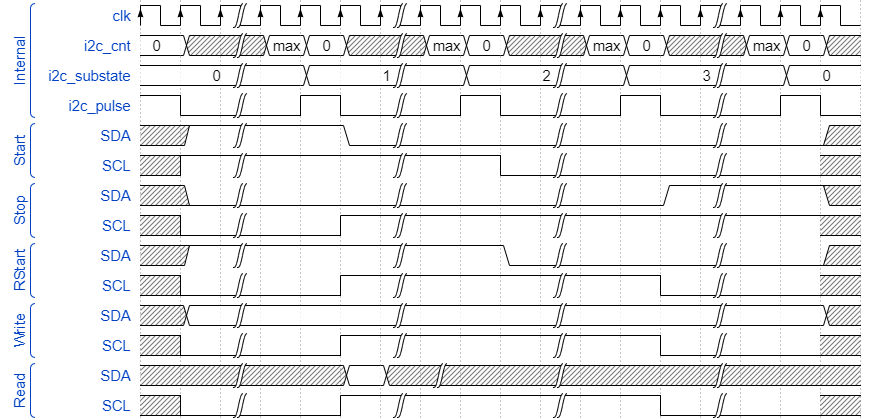
#### Status Register

#### General Internal Registers

#### Data Shift Register

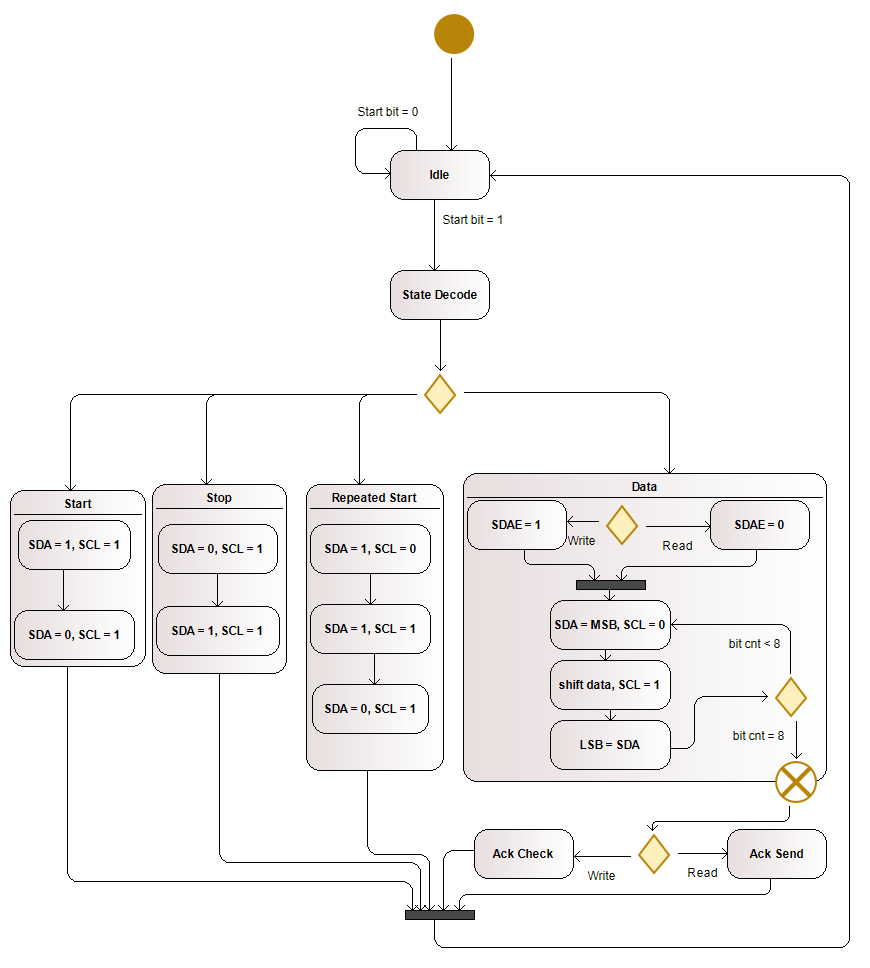
Shift register stores 8 bits and shifts the MSB onto the I2C bus every I2C clock cycle or shifts the bit on the I2C bus into the LSB every I2C clock cycle.

## Clocking



clk\_i2c\_write and clk\_i2c\_read will be combined into a single signal where the falling edge will trigger the read logic. This is to reduce the signal count and I guess to maybe reduce readability.

## I2C Event State Machine



Transmit and Receive can write to SDA from the MSB before clock high and read from SDA to the LSB during clock high. These can happen during every transaction. The result is that during transmit, the reading circuitry will record the data just exported back into the data register, basically a circular register.

# APB Interface

## Manual Control Registers

The internal registers are passed directly through to the APB interface

## Automated Polling Registers

Additional registers are available to provide automated I2C packets.

e.g. The LiteOn LTR-329ALS-01 Optical Sensor has 4 data registers that must be read in a certain order to receive the complete data for its 2 optical channels. This sequence consists of 12 bytes that must be read and written to the I2C bus and is likely performed in the same order many times while the system is running. Automated Polling Registers allow this sequence to be stored within the I2C Core and offloaded from the system controller. An interrupt will be triggered to indicate when the sequence is complete and the data may be read or updated.

### Automated Control Register

0x10

|  |  |  |
| --- | --- | --- |
| Bit | R/W | Function |
| 7 - 2 | R | 000000 – 111111 : Sequence counter capable of indicating up to 64 registers. Sequences longer than this are able to be configured in which case this becomes inaccurate. |
| 1 | W | 0 : Automatic polling is idle.  1 : Initiate an Automatic Sequence. |
| 0 | W | 0 : Automatic polling is off.  1 : Automatic polling is enabled.  \* If set to 1, an interrupt signal may be used to trigger the automatic sequence |

### Universal Register

0x80 start. Configurable.

|  |  |
| --- | --- |
| Bit | Function |
| 7 | Selects Automated Sequence registers |
| 6 | 0 : register data, bits 7 - 0  1 : register command, bits 9 - 8 |
| 5 - 0 | Automated Sequence Address |

Each Universal Register may contain an I2C slave address, an I2C slave register address, or I2C slave register data.

|  |  |  |
| --- | --- | --- |
| Bit | R/W | Function |
| 9 - 8 | W | 00 : No Operation, used to identify unused registers.  01 : bits 7-0 identify special conditions such as START, STOP, etc.  10 : data in bits 7-0 are to be written to the I2C bus  11 : data in bits 7-0 are to be read from the I2C bus |
| 7 - 0 | W | Bits to be read or written to the I2C bus where 7 is the MSB  If bit 9 = 1:  0b00000001 = START  0b00000010 = STOP  0b00000011 = REPEATED START |