**I2C Interface**

1. Introduction

I2C Interface is a component with 2 1-bit line busses which makes possible the communication between I2C Protocol and Wishbone Protocol. These lines are SCL(Serial Clock) and SDA(Serial data) that communicates with I2C device and CLK, RST, DATA\_O, DATA\_I, ACK, WE, INTA which communicates with Wishbone Device(Master).

1. Features

• Compatible with I2C specification

– Multi-master operation

– Clock stretching and wait state generation

– Interrupt flag generation

– Arbitration lost interrupt, with automatic transfer cancellation

– Bus busy detection

• Compliant with WISHBONE specification

* Compliant WISHBONE Classic interface

– All output signals are registered

– Two-cycle access time

1. Top level block diagram

I2C Device

I2C INTERFACE

RST

WE

CLK

DATA\_O

ACK

DATA\_I

INT

WISHBONE DEVICE

SCL

SDA

1. Architecture

The interface will contain the following major blocks:

* WRITE\_FSM, a FSM for sending information to I2C from Wishbone
* READ\_FSM, a FSM for receiving information from I2C to Wishbone
* INTERNAL\_REG which contains the following internal registers(communicate with Wishbone interface):
* PRESCALE\_REG is used to prescale the SCL, and also the FSM’s, based on the master(Wishbone) clock.
* CONTROL\_REG, the MSB of this register is the most critical one because it enables or disables the entire I2C core. The core will not respond to any command unless this bit is set. The other bit which is used is for memorizing the interrupt state(INTA signal).
* The TRANSMIT\_REG and the RECEIVE\_REG (same register), depends on direction of data transfer. The data to be transmitted via I2C will be stored in the Transmit Register(includes the address of Slave, with its LSB being read/write status), while the byte received via I2C is available in the Receive register.
* The STATUS\_REG and the COMMAND\_REG share the same address. The Status Register allows the monitoring of the I2C operations, including ACK signal (acknowledge), busy state of the I2C line, arbitration lost, a flag which sends an interrupt if the interrupt mode is enabled, while the Command Register stores the next command for the next I2C operation( start, stop, repeated start/stop, read/write, acknowledge, interrupt acknowledge). Unlike the rest of the registers, the bits in the Command Register are cleared automatically after each operation. Therefore this register has to be written for each start, write, read, or stop of the I2C operation.

Each of the above blocks contains FIFO’s for synchronization, counters, adders(for prescale).