Design and Implementation of Power Efficient OnboardCommunication Interfaces using FSM

Manoj Kumar V R[1],
[1]M-Tech student,
VLSI Design and Embedded
Systems, Bangalore Institute
of Technology, Bengaluru,
India.

Keshava A[2],
[2]Assistant Professor, Department of ECE,
Bangalore Institute of Technology,
Bengaluru, India.

ABSTRACT— Modern design prioritizes low power designs. In particular, data communication protocols like UART, I2C, and SPI, which have high power efficiency, are prevalent in wireless communication. In this study, we utilized Finite State Machines to design and implement these protocols. Cadence EDA tool was deployed for ASIC synthesis. The outcome demonstrated a notable decrease in overall power consumption.

Keywords: EDA, UART, I2C, SPI,Power-efficiency, ASIC.

1. INTRODUCTION

1.1 The **UART**, an independent Integrated Circuit (IC), is used for serial data communication between computers or ancillary devices in digital circuits, especially over long distances, due to its high reliability (Figure a). It incorporates various data transmission modes, including the full-duplex method where transmitter and receiver modules operate concurrently. UART, which includes a Baud Rate Generator (BRG), transmitter, and receiver, can transfer digital data from one device to another through parallel and serial transmission. The BRG sets the pace of the asynchronous communication. The baud rate, measured in bits per second, dictates the data transfer speed from transmitter to receiver. Before establishing the data transfer rate, transmitter and receiver must meet timing parameters. A shift register performs the conversion from parallel to serial.

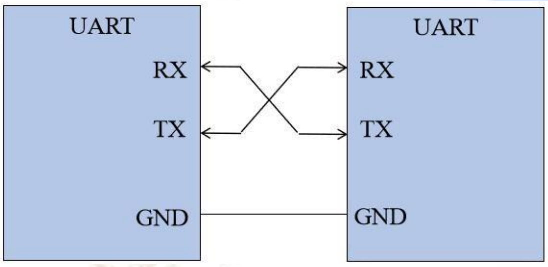
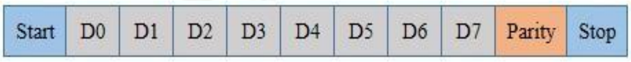


Figure.1 Block diagram-UART



Start	D0	D1	D2	D3	D4	D5	D6	D7	Parity	Stop

Figure.1.1 Data format-UART

1.2 The **Inter Integrated Circuit** (I2C) uses two bidirectional open-collector or open-drain lines: the serial data line (SDA) and the serial clock line (SCL), both equipped with resistors. The standard I2C reference design encompasses a 7-bit address space with an infrequently used 10-bit extension. Typical I2C bus speeds include 100 kbit/s standard mode and 400 kbit/s fast mode. A 10 kbit/s low-speed mode is available, with lower clock frequencies allowed. Modern I2C versions accommodate more nodes and faster speeds (400 kbit/s fast mode, 1 Mbit/s fast mode plus, 3.4 Mbit/s high-speed mode, and 5 Mbit/s ultra-fast mode), primarily applied in embedded systems rather than on PCs.

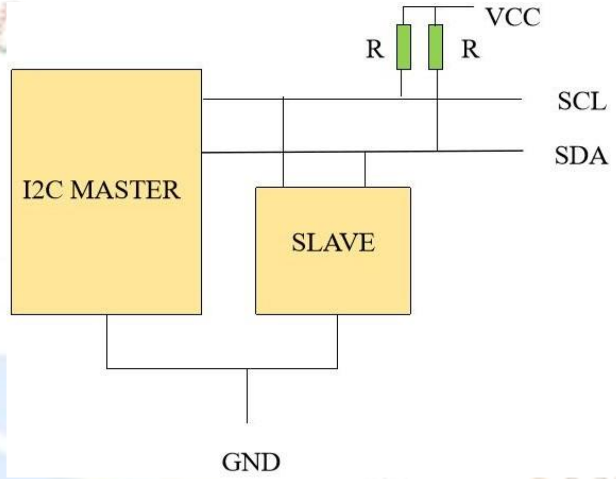
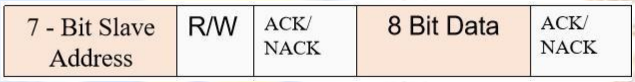


Figure.2 Block Diagram- I2C



7 - Bit Slave Address	R/W	ACK/ NACK	8 Bit Data	ACK/ NACK

Figure.2.1 Data format- I2C

1.3 The **Serial Peripheral Interface** (SPI) is a specification for synchronous serial communication interfaces that are largely utilized in embedded devices for short-distance communication.Utilizing a master-slave architecture, often with a single master, SPI devices communicate in full duplex mode.Reading and writing frames are created by the master (controller) device. Individual chip select (CS), often referred to as slave select (SS) lines, allow for the support of several slave devices through selection.

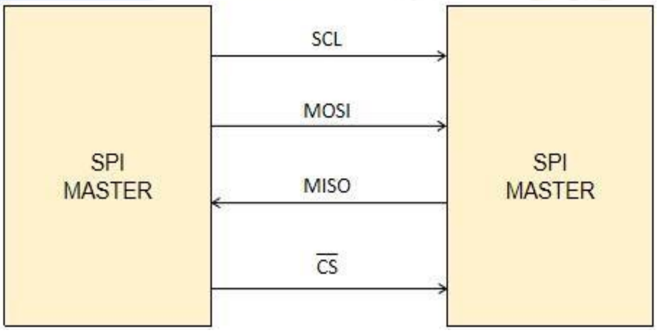


Figure.3 Block Diagram- SPI

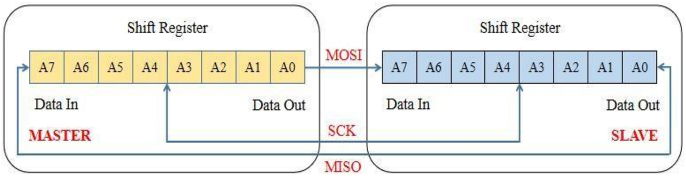


Figure.3.1 Data format flow-SPI

1.4 **FSM:** A finite state machine, also known as a finite state automaton, is a type of computation model that may be implemented using either software or hardware and is used to replicate sequential logic and some computer programmes. Regular languages are produced by finite state machines. Numerous subjects, such as mathematics, artificial intelligence, gaming, and linguistics, can benefit from the usage of finite state machines to describe problems.

2. LITERATURE REVIEW

Akshatha Kamath et al. [1] detailed their article "Design and Implementation of Power-Efficient FSM based UART", Transmitter FSM, PISO, parity generator, and baudrate generator are all included in the design of the UART transmitter. A negative edge detector, parity checker, serial in parallel out (SIPO) register, receiver finite state machine, and UART receiver are all included. It was successfully demonstrated that a high-speed UART could be designed, simulated, synthesised, and implemented using Verilog HDL.

Vivek Kumar et al. [2] described "Implementation of UART Design for RF Modules Using Different FPGA Technologies," UART implementation refers to enhancements in total power and PDP with regard to the modification of the clock periods. FSM is used for design, and 45 and 90 nm GPDK library files are used for simulation and synthesis on Cadence NCSIM Simulator and Compiler.They managed to reduce electricity consumption significantly.

Ashok Kumar Gupta et al.[3] described protocols "Design and Implementation of High-Speed Receiver and Transmitter (UART)",Verilog HDL is used to implement the design and for simulation purposes.With 180nm, this work achieved a 105.47MHz frequency.

A.Sainath Chaithanya et al.[4] conceptualized The "Design and Interfacing of I2C Master with Register and LCD Slaves",The development of the I2C protocol along with serial communication protocols helped shed light on how various I2C-capable peripherals function, and then came the construction of an I2C bus controller that interfaced with the control register of the PCA9548A. Later, the Spartan3E FPGA's I2C Master core implementation viewed the LCD, a system on chip board peripheral, as a slave.Verilog was used to design the entire module.

Shivani Mehrotra et al.[5] deviced ,"Design and Implementation of I2C Single Master on fpga using verilog" ,The I2C single master design uses a bidirectional data line, or serial data line and serial clock line. This protocol allows for numerous masters and uses just two general-purpose I/O pins to manage a network of devices. The entire module is created in Verilog and simulated in ModelSIM.

Tapaswi SJ et al.[6] designed "FSM implementation of I2C protocol and its verification using verilog", The FSM modelling approach is used to show the I2C communication protocol, and the proposed FSM's functionality is confirmed. They sought to comprehend the protocol and its fundamental hardware implementation using FSM.

Jiayi Qiang et al.[7] implemented "FPGA Implementation of SPI Bus Communication Based on State Machine Method", They described the SPI communication bus's structure and operation before using the state machine method to implement SPI bus communication on an FPGA.

Amrut Anilrao Purohith et al. [8] designed "Area Optimization using Structural Modeling for Gate Level Implementation of SPI for Microcontroller", Here, they reduced the area by reconfiguring the circuit. In comparison to the available options, the design utilised a total of 162 LUTs, which is an impressive accomplishment.

A.K. Oudjida et al. [9] delivered "Design and test of general-purpose SPI master/slave IPs on OPB bus", The paper gives a thorough explanation of current SPI Master/Slave FPGA implementations. In Verilog 2001, the entire design code is implemented, whether it is for synthesis or verification. The Master and Slave may transfer data at a rate of 71 and 75 MBPS, respectively, using the RTL coding.

Venkataramana B et al. [10] devised "ASIC Implementation of Universal Asynchronous Receiver and Transmitter using 45nm Technology," They used the Cadence RTL Compiler to synthesise the TSMC45 nm CMOS cell library and configured the system clock frequency at 50MHz. The time taken for each data bit to be transferred is 23.75 ns, and the baud rate is 42.1 Mbps (division factor is 32).

3. PROPOSED METHODOLOGY

3.1 UART Protocol FSM

3.1.1 UART Tx FSM

IDLE: This state is inactive and no operation takes place. Upon receiving a high signal, the next state is entered.
START: In this state, a start bit is added to the data frame, but it is not reflected in the receiver's output.
DATA: In this state, data is transmitted serially with a counter. Once the counter resets, the next state is reached.
STOP: This state is achieved when the counter resets. At the end of this state, the transmission is completed.

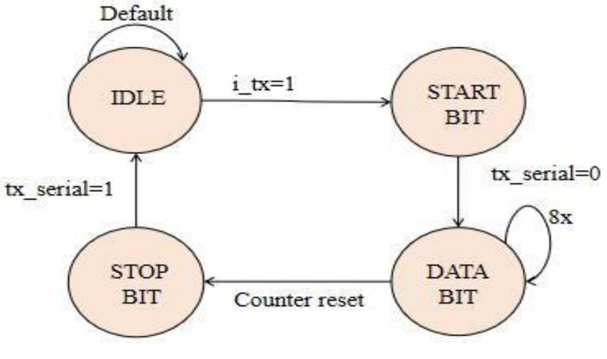


Figure. 4 Transmitter FSM-UART

3.1.2 UART Rx FSM

IDLE: In this condition, nothing is happening.
START: The start bit is acknowledged in this circumstance, but once acknowledged, it is discarded and won't be reflected at the receiver output.
DATA: In this scenario, data is received with a counter. The counter restarts and the next state is reached.
STOP: This circumstance occurs when the counter resets. After this, the reception is complete.
CLEAN UP: At this stage, the procedure is halted for 1 clock time to make sure there are no residual data.

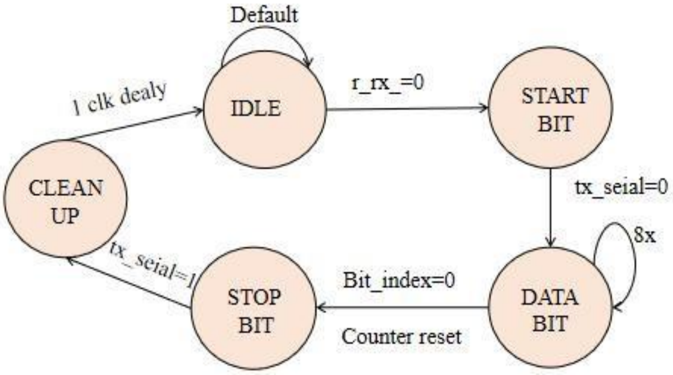


Figure. 4.1 Receiver FSM-UART

3.2 I2C Protocol FSM

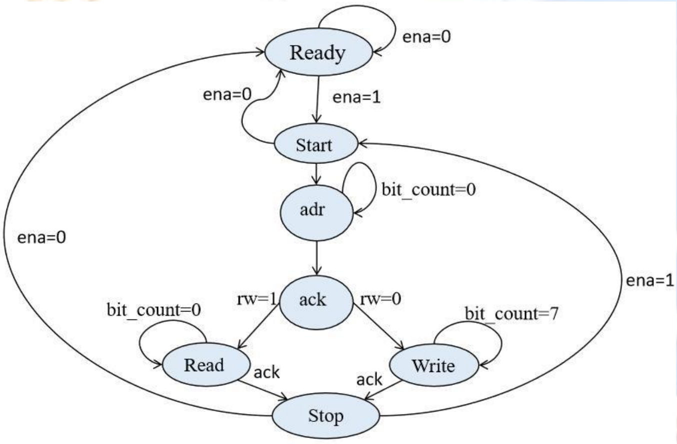


Figure. 5 I2C FSM

State 1: **Ready**: The I2C bus performs no activities at this time.Enable is low, but SCL and SDA are still high. Ena transitions to the following state if it reaches HIGH.
State 2: **Start**: The Master initiates data transmission when the next state, adr, is attained when ena is HIGH.
State 3: **Address**: In this condition, the slave is given the slave address by the master in serial form. The counter bit_cnt is used to keep track of the address bits delivered, and it changes states when it hits zero.
State 4: **Ack**: Sends an acknowledgement bit back to the master if the slave address matches the slave (in this situation, just one slave is taken into account; therefore, there is no need to match it). Now that the R/W bit has been inspected, it either enters the write state if it is LOW or remains in the read state.
State 5: **Write**: The master sends the slave the 8 bit data that needs to be delivered in state 5. The slave thanks the master for the data after receiving it.
State 6: **Read**: During this phase, the slave's 8-bit data is read by the master. Reading the data results in an acknowledgement being given.
State 7: **Stop**:After the data has been transmitted, a STOP bit is given.SCL is high, and SDA ranges from low to high. the "Master sends"To break the connection, the master transmits a STOP bit. To break the connection, the master transmits a STOP bit.

3.3 SPI Protocol FSM

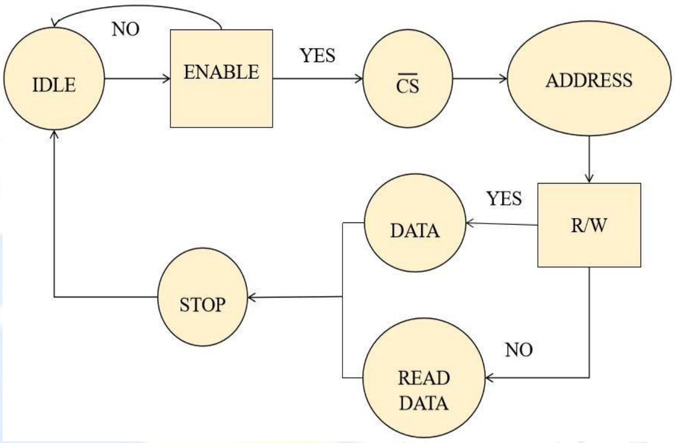


Figure. 6 SPI FSM

State 1: **Idle**: The SPI is not operating.Ena moves on to the next state if it is HIGH.
State 2: **Enable**: The Master checks for chip select while enable is HIGH.
State 3: **Chip Select**: Control moves on to the following state if it is chosen.
State 4: **Address**: Once CS is verified, the master starts the data transmission.
State 5: **Read/Write**: The following state is entered based on the R/W status.
State 6: **Data**: Write data is sent during this state, and then it stops.
State 7: **Read** Data: Data is read in this state, and then it stops.
State 8: **Stop**: this condition is reached following a counter reset.

4. RESULTS AND DISCUSSION

The FSMs for the three protocols under consideration are put into practise, and synthesis reports are produced. Cadence Genus EDA is used for the synthesis, and results of simulation and synthesis are obtained for the design. The power reports of the protocols built on the FSM have been improved.These are shown in the following figures, Figure 7, 8,9,10,11 and 12.

4.1 UART Simulation and Synthesis Results

The UART FSM used a baudrate of 9600 at 25MHz, the protocol successfully transferred an 8'bit data-8'h 3A.

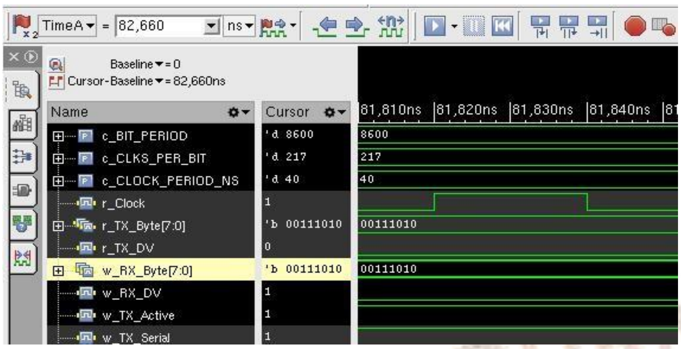
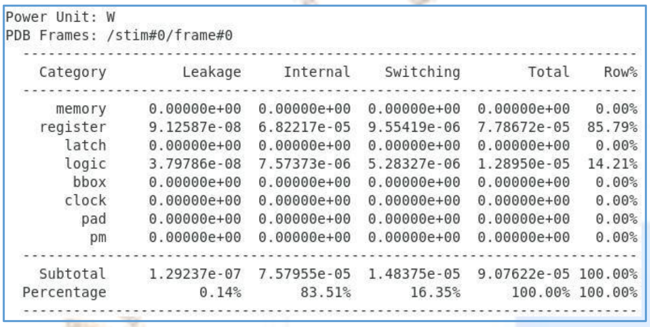


Figure.7 Simulation waveform of UART



```
Power Unit: W
PDB Frames: /stim#0/frame#0
  -------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  -------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     9.12587e-08  6.82217e-05  9.55419e-06  7.78672e-05  85.79%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     3.79786e-08  7.57373e-06  5.28327e-06  1.28950e-05  14.21%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  -------------------------------------------------------------------------
    Subtotal     1.29237e-07  7.57955e-05  1.48375e-05  9.07622e-05 100.00%
  Percentage           0.14%       83.51%       16.35%      100.00% 100.00%
  -------------------------------------------------------------------------
```

Figure.8 Power Report of UART

4.2 I2C Simulation and Synthesis Results

The I2C FSM successfully transferred an 8'bit data-8'h CC and 8'h FC to address 1010101 and 1110101 respectively.

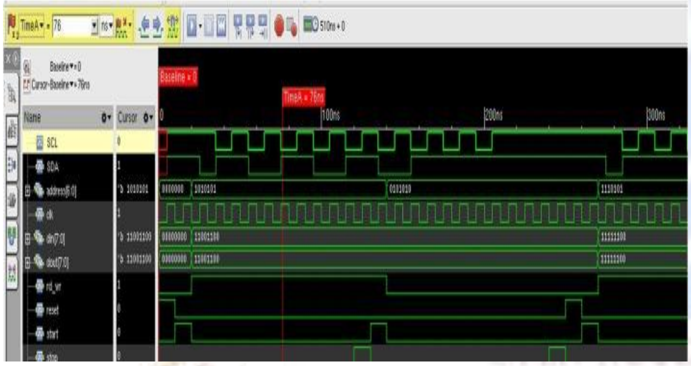
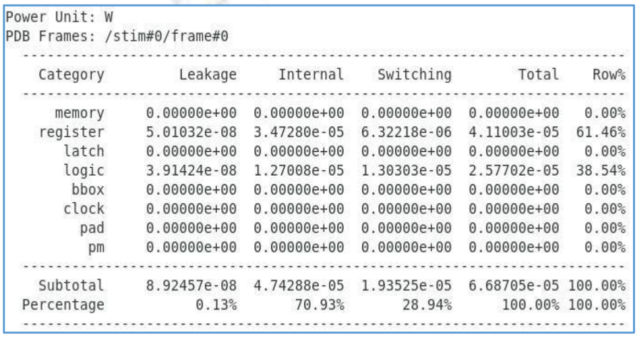


Figure.9 Simulation waveform of I2C



```
Power Unit: W
PDB Frames: /stim#0/frame#0
  -------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  -------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     5.01032e-08  3.47280e-05  6.32218e-06  4.11003e-05  61.46%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     3.91424e-08  1.27008e-05  1.30303e-05  2.57702e-05  38.54%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  -------------------------------------------------------------------------
    Subtotal     8.92457e-08  4.74288e-05  1.93525e-05  6.68705e-05 100.00%
  Percentage           0.13%       70.93%       28.94%      100.00% 100.00%
  -------------------------------------------------------------------------
```

Figure.10 Power Report of I2C

4.3 SPI Simulation and Synthesis Results

The SPI FSM successfully transferred an 8 bit data-8'h 8A and 8'h AB successive iterations.

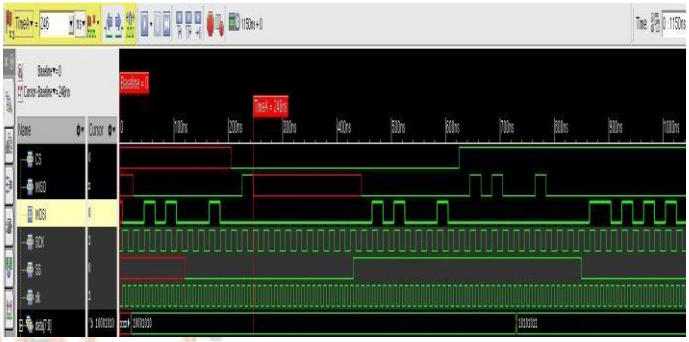
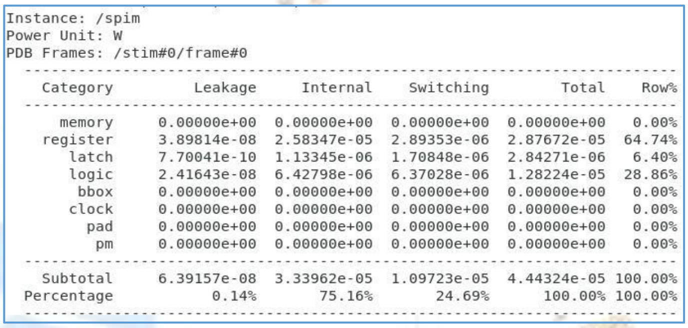


Figure.11 Simulation waveform of SPI



```
Instance: /spim
Power Unit: W
PDB Frames: /stim#0/frame#0
  -------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  -------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     3.89814e-08  2.58347e-05  2.89353e-06  2.87672e-05  64.74%
       latch     7.70041e-10  1.13345e-06  1.70848e-06  2.84271e-06   6.40%
       logic     2.41643e-08  6.42798e-06  6.37028e-06  1.28224e-05  28.86%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  -------------------------------------------------------------------------
    Subtotal     6.39157e-08  3.33962e-05  1.09723e-05  4.44324e-05 100.00%
  Percentage           0.14%       75.16%       24.69%      100.00% 100.00%
  -------------------------------------------------------------------------
```

Figure.12 Power Report of SPI

Power consumption in any IC designs, circuitry, interfaces etc are crucial. It is vital to reduce the overall power consumption in such designs which is also the aim of this project. Protocols with low-power consumption can increase the durability and effectiveness of battery-operated devices, aid in network scalability, and lessen the overall environmental impact. This draws the need of reduction in power usage.

Table.1 Comparison of power reports of Proposed Design and Existing Design

Protocol	Power (µW) Proposed Design	Power (µW) [Ref]
UART(Transceiver)	90.7	1492 [13]
I2C(slave)	66.8	157.1 [11]
SPI(master)	44.4	81.42 [12]

With reference to the Table.1 that is tabulated using the ASIC synthesis reports obtained after implementation of the protocols show that the designed protocols i,e. UART, I2C and SPI consumed the power of 90.7 µW, 66.8 µW and 44.4 µW respectively. These values are significantly lower compared to existing designs referred.

5. Conclusion and Future Scope

5.1 Conclusion

The new FSM-based protocols are designed using Verilog HDL, and then synthesised using Cadence Genus Synthesis Solution. The UART, I2C, and SPI are implemented by the Finite State Machine (FSM). According to the synthesis results for the FSM-based design protocols, less overall power is consumed than in earlier designs. FSMs are used to implement the suggested design, which reduces overall power consumption.

5.2 Future Scope

These protocols have been used in a significant portion of communications.Lower power usage is given priority in this work so that other criteria and parameters can be taken into account in future studies.Delay may be taken into consideration and can be made using a variety of other methods in future chip area improvements. In Future research on power-efficient protocols for RFID, 5G, and edge computing is possible.

REFERENCES

[1]. Akshatha Kamath, Tanya Mendez, Ramya S, Subramanya G Nayak, "Design and Implementation of Power-Efficient FSM based UART", 2022 , Journal of Physics: Conference Series IOP Publishing doi:10.1088/1742-6596/2161/1/012052

[2]. Vivek Kumar, Aksh Rastogi and V.K.nTomar, "Implementation of UART Design for RF Modules Using Different FPGA Technologies," Int. Conf. on Futuristic and Sustainable Aspects in Eng. and Technology(FSAET 2020). DOI:10.1088/1757-899X/1116/1/012131.

[3]. Ashok Kumar Gupta, Ashish Raman, Naveen Kumar and Ravi Ranjan, "Design and Implementation of High- Speed Receiver and Transmitter (UART)," 2020 7th Int.conf.on Signal Processing and Integrated Networks (SPIN), Noida, 27 - 28 Feb. 2020, pp. 295-300.

[4]. A.Sainath Chaithanya, D.Sindhuja, D.Bhavana, P.Vennela, "Design and Interfacing of I2C Master with Register and LCD Slaves", International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958 (Online), Volume-9 Issue-4, April, 2020.

[5]. Shivani Mehrotra1, Nisha Charaya, "Design and Implementation of I2C Single Master on fpga using verilog", Progress In Science and Engineering Research Journal ISSN 2347-6680 €

[6]. Tapaswi SJ1, Dr.Kiran V, "FSM implementation of I2C protocol and its verification using verilog", International Journal of Advances in Engineering and Management (IJAEM) Volume 3, Issue 9 Sep 2021, pp: 1058-1060 www.ijaem.net ISSN: 2395-5252.

[7]. Jiayi Qiang1, Yong Gu2 and Guochu Chen, "FPGA Implementation of SPI Bus Communication Based on State Machine Method", ISPECE 2019, doi:10.1088/1742-6596/1449/1/012027.

[8]. Amrut Anilrao Purohit, Mohammed Riyaz Ahmed, R. Venkata Siva Reddy, "Area Optimization using Structural Modeling for Gate Level Implementation of SPI for Microcontroller", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-9 Issue-1, November 2019.

[9]. A.K. Oudjida, M.L. Berrandjia, A. Liacha, R. Tiar, K. Tahraoui & Y.N. Alhoumays,"Design and test of general-purpose SPI master/slave IPs on OPB bus", DOI: 10.1109/SSD.2010.5585592.

[10]. Venkataramana B, Srikanth Reddy P, Kishore Kumar P, and Sivasankaran K, "ASIC Implementation of Universal Asynchronous Receiver and Transmitter using 45nm Technology," Int. J. of Eng. and Technol. (IJET), 2013. 5(3), pp. 2765-2770.

[11]. Imran Ali, Sung Hun Cho, Dong Gyu Kim, Muhammad Riaz Ur Rehman and Kang-Yoon Lee,"A Design of Ultra Low Power I2C Synchronous Slave Controller with Interface Voltage Level Independency in 180 nm CMOS Technology",College of Information and Communication Engineering ,Sungkyunkwan University, Suwon, Korea, ISOCC 2017,978-1-5386-2285-8/17/$31.00 ©2017 IEEE

[12]. Arjumanth Farraj R, Kiran V. Design, verification and testing of comprehensive serial peripheral interface. International Journal of Research and Review. 2022; 9(11): 46-53. DOI: https://doi.org/10.52403/ijrr.20221107

[13]. Badam Suresh, P. Teja Reddy, V.S.V. Srihari and S. Sivanantham, "ASIC Implementation of Low Power Universal Asynchronous Receiver Transmitter", World Applied Sciences Journal 32 (3): 472-477, 2014,DOI: 10.5829/idosi.wasj.2014.32.03.261