Dynamic Reconfiguration of Bluetooth Baseband Bitstream Processing

John Esquiagola, Guilherme Ozari, Marcio Teruya, Marius Strum, Wang Chau
Laboratório de Microeletrônica - Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, Trav. 3, Nro 158. Cidade Universitaria
São Paulo - Brazil
{jedward, galmeida, teruya, strum, wang}@lme.usp.br

ABSTRACT
Field Programmable Gate Arrays (FPGAs) provide an attractive solution to implement dynamically reconfigurable architectures using commercial FPGA devices. The feasibility of dynamic reconfiguration of Virtex FPGAs has been established by a large number of case studies. In this paper, we describe the structure and implementation of a Dynamically Reconfigurable Bluetooth Baseband System using Xilinx Virtex FPGAs. The system implemented was partitioned in modules for transmission and reception of data packets, and a language of high level of abstraction such as SystemC was utilized to describe the system, this SystemC code is translated into VHDL description, after the modular design methodology is applied to obtain the partial and total bitstreams of the system.

1. INTRODUCTION
The concept of reconfigurable computing [1] was established many years ago, but has only recently become achievable due to the availability of hardware offering the kinds of services necessary to apply them to “real-world” scenarios. Dynamic reconfiguration of FPGAs has recently become viable with the introduction of devices that allow high speed partial reconfiguration, e.g. the Xilinx Virtex Series[2]. A dynamic reconfigurable device allows a portion of a user design to remain in place while another portion is being updated. The approach of applying reconfigurable logic for data processing has been demonstrated in some areas such as video transmission, image-recognition and various pattern matching operations [3]. Wireless communication systems become increasingly more computational intensive and demand for higher flexibility. The realization of these systems on reconfigurable hardware offers a good balance for these requirements.

Specification at high level of abstraction is possible in environments such as SystemC. SystemC is an emerging standard modeling platform based on C++ that supports design abstraction at the RTL, behavioral and system level [4, 5]. Our goal is to implement a dynamically reconfigurable baseband module of a Bluetooth controller using a commercial Virtex-Xilinx FPGA.

In section 2, the Bluetooth standard is described. Section 3 details the system architecture, showing how the system is partitioned to obtain the dynamic reconfiguration. The methodology of implementation is described in section 4. Section 5 presents the results obtained until the moment. Section 6 presents our conclusions and future work.

2. BLUETOOTH STANDARD
Bluetooth is a standard for short distance wireless communications developed by the Bluetooth Special Interest Group (SIG) [6]. The Bluetooth Specification was developed to substitute cables connecting portable or desktop devices and build low-cost wireless networks for mobile and portable devices. The Bluetooth Stack in figure 1., goes from high level application layer to the low level radio frequency layer, the baseband layer has been implemented in this work. The Bluetooth standard operates at 2.4 GHz in the ISM band (Industrial, Scientific, Medicine) with GFSK modulation (Gaussian Frequency Shift Keying). The Bluetooth devices sharing the same channel from a network called piconet, with a single unit acting as a master, the other units acting as slaves. Up to seven slaves can be active in the piconet.

The standard defines two types of link between master and slaves: Synchronous Connection-Oriented link (SCO), and Asynchronous Connection-Less link (ACL).

2.1 Baseband Bitstream Processing
Channel coding of data before transmission through a RF channel is essential to protect that data against an imperfect channel. The bitstream processing blocks of the baseband unit are: FEC, FHEC, WHITE, CRC. They are connected in such a way as to allow data to stream through the blocks continuously without any buffers between them. The "Stream Processor" has been designed to control each function block according to the type of

---

1 This work was supported by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) grant.
packet. In terms of error detection, a packet is partitioned into three parts: access code, header for HEC, and payload for CRC. The whitening block randomizes the data and minimizes DC bias. It reduces a long sequence of zero’s or one’s by mixing data bit-stream and pseudo-random bits [6].

The TX/RX bit-stream processing varies depending on the received packet. At this time, RX processing is different from TX processing. According to figure 1, the HEC block is necessary for the packet header then the header bits pass to the scrambling process using a word of whitening. Then FEC 1/3 codification is applied. In the receiver, inverse processes are carried through. The figure 2 shows the process for the payload, where similar processes are executed, concurrently with the processes defined for the packet header. The process of whitening and de-whitening is only obligatory for each payload, the other processes are optional and depend on the type of packet and the enabled mode.

The overall system architecture of the Bluetooth Baseband Bitstream Processing Unit to be designed includes the bitstream processing blocks (CRC, FEC, FHEC, WHITE), the Stream Processor and the Packet Buffer.

The CRC module performs error detection and correction on the packet payload. It generates polynomial CRC for data transmission and makes the verification on data reception. For each new packet of data, the internal LFSR of the module must be initialized by the Stream Processor. The LFSR is preloaded with the slave upper address part (UAP). Since the UAP is an 8 bits value, it is loaded into the 8 least significant bits. The other bits are set to zero. The CRC code is appended to the information.

The FEC module performs the generation and verification of FEC 2/3 for the payload. It is used to reduce the chances of getting corrupted information. A polynomial generator produces the code of Hamming (15,10). The polynomial generating circuit is initialized with zero, then for each 10 bits of data, 5 parity bits are generated and appended to 10 bits word before being transmitted.

The FHEC module implements FEC 1/3 and HEC in just one module. These two processes are developed for the processing of the packet header. The FEC 1/3 code is used to protect the packet header. This code is generated for the header of the entire packet, including the HEC code. The HEC code is generated by 3 simple repetitions of the header bits. The 18 bits header is essentially coded into a header of 54 bits.

The Whitening module performs the scrambling and de-scrambling of the header and payload for both the RX and TX directions. The whitening of data minimizes the DC polarization in the packet. The packet header and payload are scrambled before they are codified (FEC). The received packet is first decoded (FEC) and then unscrambled.

The Stream Processor does the treatment of the incoming data and prepares the data to be transmitted. It instantiates the modules and defines the communications among the Packet Buffer, CRC, FEC, FHEC and WHITE. It is composed of a state machine that executes several tasks: Connection between the modules and the Packet Buffer through one multiplexer, initialization of the modules, reading of the packet type to be transmitted, transmission of the payload, reception of the header, reception of the payload.

4. DYNAMIC SYSTEM IMPLEMENTATION

For the static case and depending on the size of the system, a suitable FPGA could be chosen. In order to reduce the size of the hardware, the technique of “Virtual Hardware” is used. This technique uses the capacity of partial and total dynamic reconfiguration of the current FPGAS, allowing a great circuit to be partitioned into small sub-circuits.

In this work, the Virtex-II FPGA from Xilinx [2] has been used because it allows partial and total dynamic reconfiguration. The size of the devices in the Virtex-II family varies. The architecture is based on an array of Configurable Logic Blocks (CLBs) interconnected through a general routing matrix. Each CLB contains a routing matrix and four slices. Each slice consists of two Look-Up Tables (LUTs) and two flip-flops. In [7], Xilinx defines a design flow for digital designs implementation using partial reconfiguration.

4.1 The dynamic architecture

The Stream Processor module, described in section 3.5,
was developed for a static implementation. It had to be modified to satisfy the requirements of the dynamic reconfiguration. This new “Stream Processor” module has a new state machine that performs the reconfiguration process and it only has the white module instantiated, due to this module is used for the header and payload process. Eight xilinx bus macros were placed at the interface. Bus macros are used to assure data communication between fixed modules and reconfigurable ones. As bus macros have only 4-bits width, we instantiated as bus macros as required to transmit all the required signals.

![Figure 3. Partial Dynamic Architecture of Baseband Unit](image)

Based on the knowledge of the design architecture and the use of each module in time, one can know which part of the system to dynamically load, and under which conditions. According to the figures 1 and 2, the header bits and payload bits require different modules and the execution flow is sequential, first the header and then the payload. The header requires FHEC and WHITE, while the payload requires CRC, WHITE and FEC. The state machine implemented in the Stream Processor module is modified so that it can initialize the modules in different instants of time: t1 for the FHEC and t2 for CRC and FEC. The whitening module is instantiated inside the Stream Processor because it is used for both, the header and the payload. Figure 3 shows the partial dynamic architecture for the time t1.

### 4.2 Adapted Design Flow

The aim of this section is to introduce a methodology for FPGA dynamic reconfiguration. SystemC non–RTL is adopted to capture the design. The adapted design flow shown in figure 4 is based on Synopsys and Xilinx CAD tools. The Bluetooth Golden Model was developed at Synopsys. The non-RTL SystemC model of each module was verified using a functional verification testbench with the golden model as reference and SystemC libraries.

![Figure 4. Adapted Design Flow](image)

Then the non-RTL SystemC model is refined to create the RTL SystemC model. The functional verification testbench is used once again to validate the model. Then, the RTL SystemC was translated to VHDL using the Synopsys Design Compiler. The verification of this model was done using SystemC-VHDL/Verilog cosimulation.

After the cosimulation process, the Xilinx Modular design methodology [7, 8] is applied to implement the dynamic architecture.

## 5. RESULTS

Two solutions were implemented: static system and dynamic system. The FPGA Virtex-II XC2V1000 has been utilized as the reconfigurable device. This device is organized as a set of arrays of 64 columns by 80 rows. The dynamic system implementation in figure 5 shows the time t2 context that includes the Stream Processor, CRC and FEC modules.

Table 1 shows the results after the synthesis, placement and routing processes for all modules separately. The tool used is Xilinx ISE 6.3i.

<table>
<thead>
<tr>
<th>Module</th>
<th># Slices</th>
<th># Flip-Flops</th>
<th># 4-input LUTs</th>
<th>Max. Clock Freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>17</td>
<td>19</td>
<td>32</td>
<td>210 Mhz</td>
</tr>
<tr>
<td>FEC</td>
<td>80</td>
<td>62</td>
<td>148</td>
<td>124 MHz</td>
</tr>
<tr>
<td>FHEC</td>
<td>56</td>
<td>46</td>
<td>103</td>
<td>162 MHz</td>
</tr>
<tr>
<td>WHITE</td>
<td>10</td>
<td>12</td>
<td>18</td>
<td>211 MHz</td>
</tr>
</tbody>
</table>

Table 1. Results of the synthesis process
The system for the static case uses 426 slices, occupying 10% of the FPGA resources. For the dynamic case it uses 315 slices for the $t_1$ context, and 402 slices for the $t_2$ context. According to the Virtex-II platform user guide, the maximum configuration bitstream programming rate is 66MHz with handshaking and 50MHz without handshaking. Using the SelectMap interface, one byte at a time can be written. The bitstream size for a complete configuration of the XC2V1000 device is 510,324 bytes. The configuration download time at 50MHz becomes 10.2ms. Actually the reconfigurable part of the system implemented has 8 columns, but this can be reduced to 4 columns to minimize the reconfiguration time. This minimal number of columns is suggested by Xilinx modular design methodology. The size of partial bitstream obtained is 22Kbytes. Using clock of 50MHz, the time of reconfiguration is 22kytes* 20ns = 440µs.

In the Bluetooth system, a complete packet transmission occurs during one time slot of 625µs. Our system has three processing times. The first $t_h$ is the time for the header processing. The second $t_{rec}$ is for the reconfiguration, and the last time $t_p$ is for the payload processing. Like the time of reconfiguration is 440µs, then $t_h + t_{rec} = 185µs$. According to the synthesis process, the performance of the system reaches 100Mhz approximately. With this system clock the values of $t_h$ and $t_p$ are realistics.

6. CONCLUSIONS

This work presents a dynamic reconfiguration implementation of a Bluetooth Baseband Bitstream Processor. The system implements the transmission and reception of Bluetooth packets admitting a partition of this application in time that elapses of the execution in series of the header and payload process for one Bluetooth packet. Furthermore, the ongoing implementation process using the Xilinx partial reconfiguration within the modular design flow is described. The current state of the implementation and the major aspects of the design process are presented and discussed. The units which are not reconfigured (Stream Processor, Packet Buffer, and Whitening) build up a fixed module. Other units (CRC, FEC, FHEC, dummy) build up the reconfigurable module. Xilinx bus macros are used to assure the control and the data flow between the fixed and the configurable modules.

The synthesis process results in 426 slices for the static system, and 315 and 402 slices for the dynamic case ($t_1$ and $t_2$ contexts). The device XC2V1000 is used to implemented the design. The partial bitstream obtained was 22Kbytes, and the reconfiguration time utilizing a 50MHz clock was 440µs. According to the synthesis process, the performance of the system reaches 100MHz approximately. As the Bluetooth time slot is 625µs, there is enough time to execute the header plus payload processes for this solution.

The main focus of the ongoing work is the implementation of the complete model of the Bluetooth Baseband system that includes some additional blocks: Packet Processor, Correlator and one LEON µprocessor. These modules will maximize the complexity of the system. The impact of reconfiguration time penalty on the performance due to these extra functional units has to be investigated. All these parameters will determine the final performance of the system.

7. REFERENCES


