Design of an ARM Based System-on-a-Chip for Pay Phones

Juan C. Díaz Jacobo Riesco Pierre Plaza

Telefónica Investigación y Desarrollo
Emilio Vargas 6, 28043 Madrid, Spain

1. Abstract

This paper describes the design of a microcontroller intended to be a system-on-a-chip solution for the Spanish Pay Phone System. The Circuit is based on the present TMPLUS Pay phone equipment, deployed in Spain, South America, and some other countries all over the world.

The design has been built around the ARM microprocessor using Intellectual Property (IP) blocks from ARM, from other IP providers, and custom block developed at Telefónica I+D.

The available ARM tools have allowed a Hardware and Software co-design approach, reducing the global design time.

1.1. Keywords

Pay Phones, System on a Chip, Embedded ARM, IP design reuse.

2. Introduction

The terminals in the pay phone Spanish network have to be adapted to the growing needs of a market that is suffering important transformations: the European Telecom market deregulation, the new internet era and attractive innovative multimedia services. These factors were the driving forces behind the need of designing a custom microcontroller ASIC. Telefónica Investigación y Desarrollo (TID) designed in 1989 a pay phone system now deployed not only in Spain but in many different countries (Australia, Argentina, Chile, etc…). Almost ten years later, this design is still valid, but cannot provide the new features that are needed to face competition and provide enhanced services.

In 1997 TID has undertaken a new pay phone terminal design without modifying the terminal's appearance, increasing the performance of the system by substituting the old 8 bits microcontroller used in the first version (8051) with a 16 bits microcontroller (8051XA), but maintaining the basic principles and characteristics of the system. Assuming that this new version is still limited for facing the requirements of the future public communications, TID is now working on the next version of the pay phone system. The chosen solution had to apply a completely new approach: the design will have to contain a customized microcontroller profiting from the integration capabilities of the new deep sub-micron technologies and the availability of attractive cells on the silicon vendor libraries.

Taking into account that TID is not a microprocessors designer company, and the obvious time schedule limitations on a project like this, the best solution in terms of time and performance is the use of intellectual property (IP) modules. Furthermore, the design of the circuit has been developed under the ESPRIT Project 20724 (TOMI). The aim of the TOMI project is to provide the missing methods and tools that makes practical the OMI concept of supercell based system design [1]. The µPP microcontroller is used as test vehicle for the evaluation of the tools developed in the project.

3. The "TM PLUS" Pay Phone System

The "TM", initials from the Spanish "Modular Telephone" [2], is a design widely proven in the terminals deployed in Spain, South America and other countries in the world; this system has been taken as starting point for the design of the µPP microcontroller. The TM Plus is an autonomous system, this means the only thing needed to start the operation is to connect it to a telephone line; the power needed to make the pay phones work is taken from the line. The Hardware of the TM Plus pay phone system is divided into two main boards:

- The Analog Board, which contains the circuitry, related to the power supply and the interface with the phone line. The supply system is quite complex, due

to the low current available in the line and the changing supply conditions associated to relays and mechanical devices; this power supply system is out of the scope of this paper.

- Digital Board, containing the main microprocessor and most of the peripherals.

The digital board contains two microcontrollers. The first one, called "supervisor", is an 8 bits 8051, and its main task is to manage the energy available in the system. The supervisor starts running once the system is connected to the telephone line and there is a minimum of energy in the batteries. While the telephone terminal is idle, the supervisor keeps the rest of the telephone circuitry switched off, in order to save energy; and when some events happen (someone picks up the phone, the line receives a telephone call ...), the supervisor powers up the second microcontroller, a 16 bits 8051XA, which is the main processor. When the main processor functionality is not needed, or the energy of the system goes down under a given value (due to fail on the line or the batteries), the supervisor switches off the main processor.

When the 8051XA main processor is powered up, it performs the telephone functionality, and manages the peripheral circuits: keyboard, LCD, serial lines, real time clock, pay systems (coins and cards), relays, etc. The different peripherals are connected to the 8051XA through low power EPLD circuits. When the 8051XA finishes its operation, it asks the supervisor to be switched off, always to save energy.

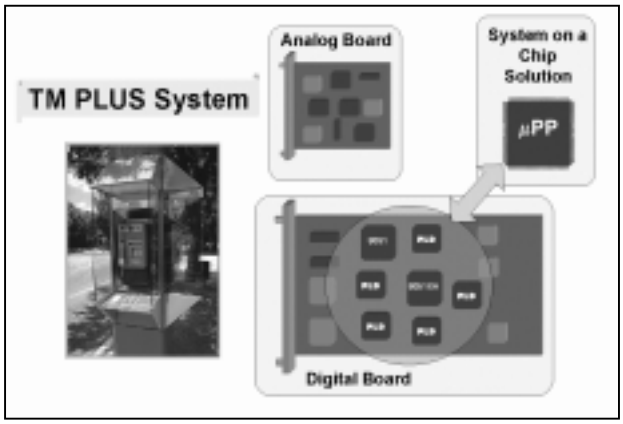

Figure 1: µPP as SoC solution for TM PLUS

The objective of the µPP microcontroller is to replace the functionality of the two microcontrollers in the digital board and to include the EPLD functionality (peripheral control) [3]. With the new microcontroller, the system will be significantly optimized in terms of cost and performance.

4. Design Constraints

The design of the µPP microcontroller must fit the following initial requirements:

- The new system has to be very modular, and the microcontroller must include the complete pay phone system functionality, so specific controllers are needed: phone keyboard, LCD Display, communications (USARTs, I^2C interface...).
- The power consumption allowed for the complete system is very small (it is battery supplied), so the microcontroller has to be very efficient, sparing as much current as possible.
- Apart from a very low power consumption in active mode, it needs to have several low consumption modes.
- In order to have a reduced design time, the microcontroller has to be based on a standard system, using IP modules and architectures.

According to these premises, an ARM microcontroller was chosen: the ARM7TDMI. This 32 bits RISC microprocessor offers, besides the best MIPS per Watt ratio in the market, the possibility of using a reduced 16 bits instruction code, oriented to embedded applications where the memory size is very limited, and a flexible bus architecture (AMBA) for developing and connecting new peripherals.

5. Circuit Architecture

The µPP circuit is based on the AMBA (Advanced Microcontrollers Bus Architecture) concept from ARM. The AMBA bus defines an on-chip communication standard for designing high performance embedded microcontrollers [4]. The definition of AMBA provides an easy way to divide the internal communication into high and low performance devices: the ASB and APB buses, acronyms for Advanced System Bus and Advanced Peripheral Bus respectively.

The ASB is a synchronous, pipelined, master-slave bus. It is intended for high bandwidth communications, typically the microprocessor with the main memory. The APB bus is asynchronous, and designed for low power and bandwidth peripherals. The connection of the APB peripherals to the microprocessor is made trough an ASB slave module called APB bridge.

The *micropack*, name given by ARM to its example AMBA system, includes all the modules required to perform the ASB and APB functionality, and some examples for ASB and APB slave peripherals; they are delivered in HDL RTL synthesisable code, with comments, design recommendations, compliance testbenches and synthesis scripts.

A representation of the internal circuit architecture is shown in Figure 2.

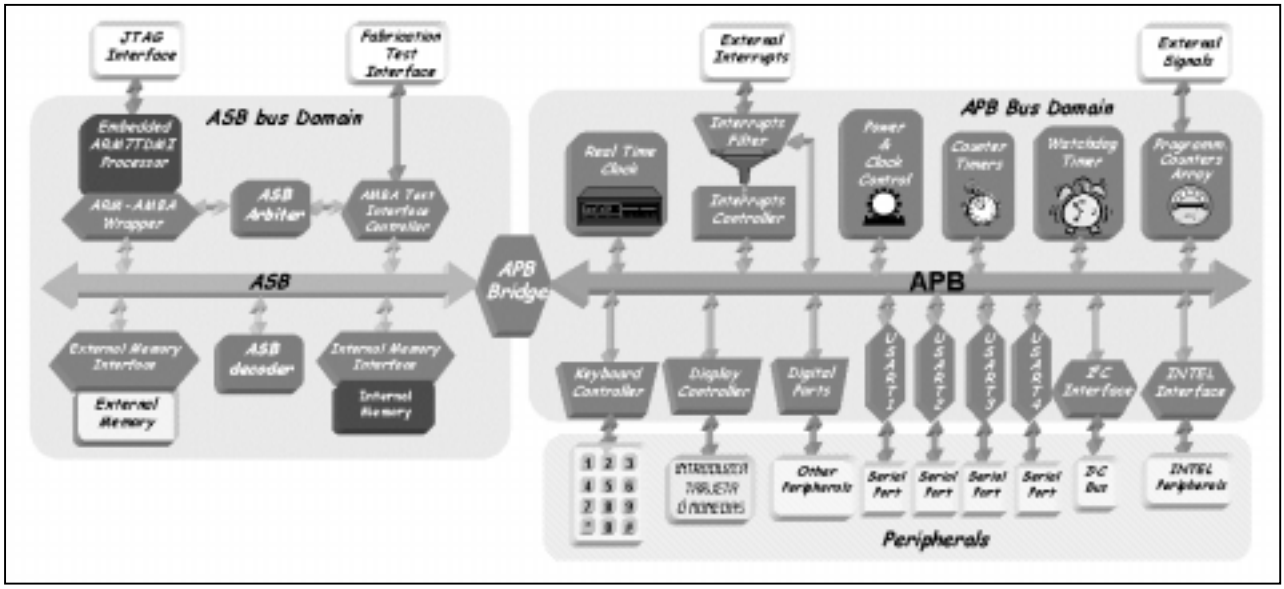

Figure 2: µPP Circuit Internal Architecture

5.1. ASB Blocks

- **ASB Arbiter**. This module controls the access to the system bus from the different master modules, therefore the ASB is a multi-master module.
- **ASB Decoder**. This module performs a centralized address decoding function.
- **ARM-AMBA Wrapper**. This module is the interface between the ARM7TDMI and the ASB bus, allowing the ARM7TDMI to become an ASB bus master.
- **Test Interface Controller**. This module is used during foundry testing as ASB master to check the internal registers.
- **External Memory Interface**. This module allows the connection of 7 external memory devices, each one with up to 256 Mbytes. The external memory characteristics (speed and bus size) are configurable using strap pins.
- **Internal Memory Interface**. The circuit contains 32 Kbytes of internal RAM memory, accessed in word mode (32 bits). The memory is accessed by the microprocessor through a dedicated ASB slave.
- **APB Bridge**. This ASB slave module provides the access to the peripherals in the APB bus, where it acts as bus master.

5.2. APB Blocks

- **Power and Clock Control**. This APB slave module can control the ASB bus clock (BCLK) frequency and manages the change to standby mode.
- **Real Time Clock**. It offers the typical functionality of a Real time clock device: date, time and alarms. It is Y2K compliant.
- **Keyboard Controller**. The circuit can manage an 8x8 passive keyboard using this interface.
- **Display Controller**. This module interfaces a standard LCD display. It contains a 256 words internal FIFO.
- **Interrupts Controller and Filter**. These two modules manage 17 internal and 10 external interrupt request signals. They perform functions like priority and polarity assignment, enabling and disabling, and noise filtering.
- **Digital I/O Ports**. This module offers the possibility of connecting up to 64 external signals: 32 outputs, 16 inputs and 16 configurable as input or outputs.
- **Counter Timers.** This module contains 3 fully programmable counter timers.
- **Watchdog Timer**. This timer acts as a standard watchdog device.
- **Programmable Counters Array**. This APB slave contains 8 counters, 8 bits wide each one, designed to generate periodic signals or monitor the frequency of input signals, depending on programming..
- **INTEL Interface**. This module allows interfacing with up to 4 external INTEL compatible devices, using an 8 bits wide Address/Data bus.
- **I^2C Interface**. The µPP I^2C Interface handles bi-directional data transmission between the microprocessor and the I^2C bus.
- **USARTs**. Four Universal Synchronous /Asynchronous Receiver/transmitters (USART) modules are connected to the APB bus.

6. Design Methodology

The first step was the identification of all the needed peripherals and functionality in the present phone system board. For all of them, a specification was developed, containing both the hardware functionality and interfaces, and the programmer's model.

This latter model was based on the ARM's reference peripherals specifications [5], in which all system elements are memory-mapped, to increase reuse and software portability. The exact definition of the memory map was not fixed initially. However, the registers inside

the hardware blocks have a fixed offset from the block's base address; this allows an independent development of the hardware and software design activities from the very beginning (see figure 3).

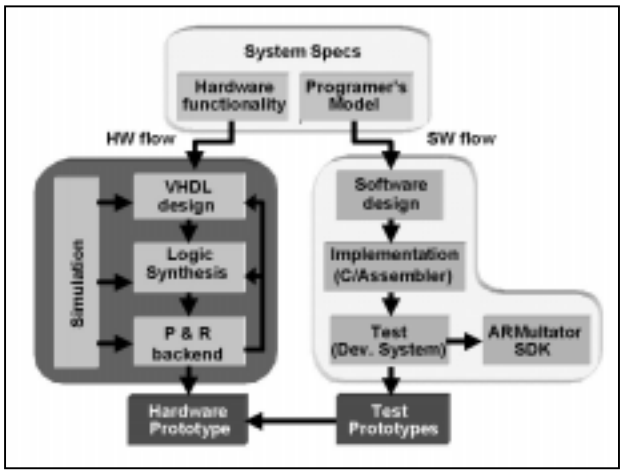

Figure 3: Design Methodology

6.1. Hardware Design Flow

The µPP hardware has a mixture of IP blocks and custom modules. IP blocks include hard, firm and soft components. The hard cells used are the ARM microprocessor and several memories generated with the ATMEL-ES2 tools; for these cells, only a simulation model and an abstract for floor-planing is provided. Firm blocks, I^2C and USARTs, are in the form of netlists, so they can be simulated and optimised during logic synthesis, but are quite difficult to modify. Soft blocks include the modules in the ARM *micropack*: both the ASB and APB blocks and some reference peripherals. They are provided in the form of synthesisable VHDL, plus Synopsys synthesis scripts. Some of them have been modified to fit the precise needs of the µPP circuit. For the maintenance, analysis, debugging and quality assurance of all the modules (both IP and custom ones), the VHDL-ICE [6] and VDS Tools have been used.

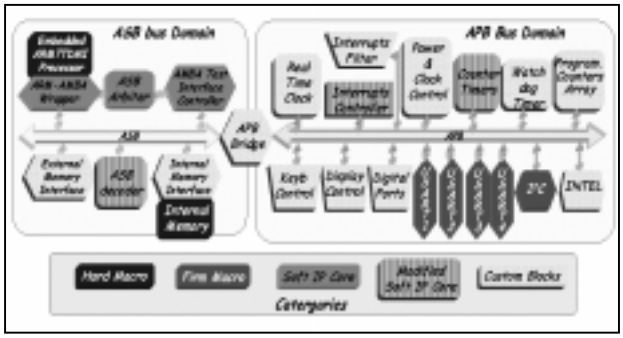

Figure 4: µPP internal blocks nature

The design flow for the custom modules followed a traditional top down methodology approach based on the use of the VHDL language and logic synthesis. First, a VHDL description of each circuit block was written and simulated using the AMBA compliance testbench suites, to perform functional as well as AMBA compliance validation of the designs. The most critical parts of the circuit, in terms of timing constraints (for example clock gating and multiplexing in the clock generation block), were described in a nearly structural, but technology independent way, using Synopsy's GTECH library components. Less critical blocks have higher level RTL descriptions.

After being validated, the soft IPs and the custom blocks were synthesised. This was a quite straightforward task thanks to the synthesis scripts available for the IP blocks. For the custom blocks, scripts were also developed to automate the process. The clock generation block required an careful static timing analysis, due to its highly asynchronous nature, to check out the constraints and the absence of glitches.

Regression tests were then performed on the netlist representation of each block and in the whole circuit, with the aid of the AMBA test interface controller (TIC) block, which provides access to the internal buses and modules through the external memory bus. Some simple programs were also run by simulating, in VHDL, a test system with the µPP model and external memories containing the ARM code. For those simulations, the circuit delays were back-annotated with pre and post-layout delays, obtained with the ATMEL-ES2 VHDL design kit tools.

The very low power requirements of the circuit forced a latch-based design in most of the blocks, that prevented the use of scan based test techniques for foundry tests. Except for the ARM core and I^2C modules that do have scan chains, all the peripheral modules have special program registers to facilitate its ad hoc testing using the AMBA test methodology (i.e. using the TIC). For the memories, BIST has been used.

The circuit Place and Route has been performed by SIDSA, the Spanish ATMEL-ES2 design support centre, using Silicon Ensemble and Clock Tree generation from Cadence. The circuit is now in fabrication at ATMEL-ES2 in Rousset.

6.2. Software Design Flow

The software development activities were run in parallel with the hardware design. First a software analysis was done, based on the existing pay phone system software. From this analysis a software architecture was built identifying the software modules to be coded in C, and the hardware drivers for the µPP peripherals, to be coded in assembler and/or C. This task is currently under development. The compiled software modules are being tested using the ARMulator software prototyping board [7].

6.3. Reuse and Optimization

Figure 4 shows the distribution of the circuit area according to the modules nature.

It can be easily inferred that more than 60% of the circuit area is occupied by cells taken form IP providers without any modification, and the area occupied by the custom modules is just 27%. It can be concluded that the design task has been noticeably eased by the reuse of IP cores, with a better confidence in the final result.

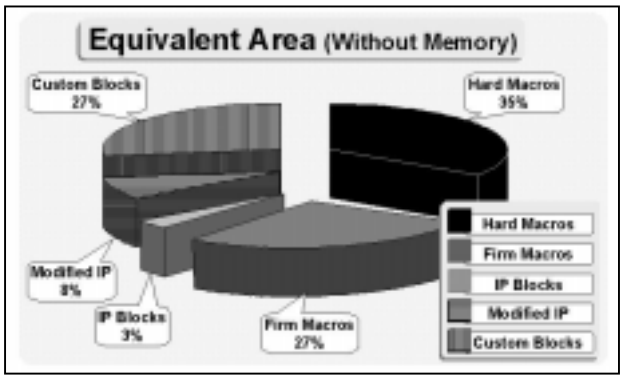

Figure 5: Area versus Modules Nature

6.4. HDL Design Management

The µPP circuit has been designed using VHDL-ICE and VDS design management environments, developed in the TOMI project; these environments, and its navigation tools, fill the gap between the circuit specification/modeling and the further synthesis, and are specially useful for evaluating the IP cores quality and synthesizability [6].

These design management tools can be used on every design stage (behavioral models, RTL models, netlists pre and post layout...), and provide a powerful way to analyze and debug both, IP cores and user designed modules, and even analyze the HDL code quality.

7. Results and conclusions

The µPP circuit is a good example of a system-on-a-chip design, where all the features of an existing system are integrated in a unique device, reducing costs and increasing reliability an system performance.

To afford the complexity of such a design in a short development time, IP blocks from different IP providers together with custom modules where used.

The ARM AMBA bus strategy and the ARM Software development Kit with the ARMulator made possible to perform in parallel the hardware and software development tasks, reducing the global system design time and offering the possibility of having a functional system as soon as the prototypes are available.

The Circuit has been implemented using the 0.5 µm ATMEL-ES2 technology (ECAT05). It contains 68K equivalent gates, 256Kbits of single port RAM, 2Kbits of dual port RAM, and 1 ARM7TDMI microprocessor. The circuit area is 72.25 mm^2, packaged in a 304 CQFP with 234 functional pins and 28 supply pins.

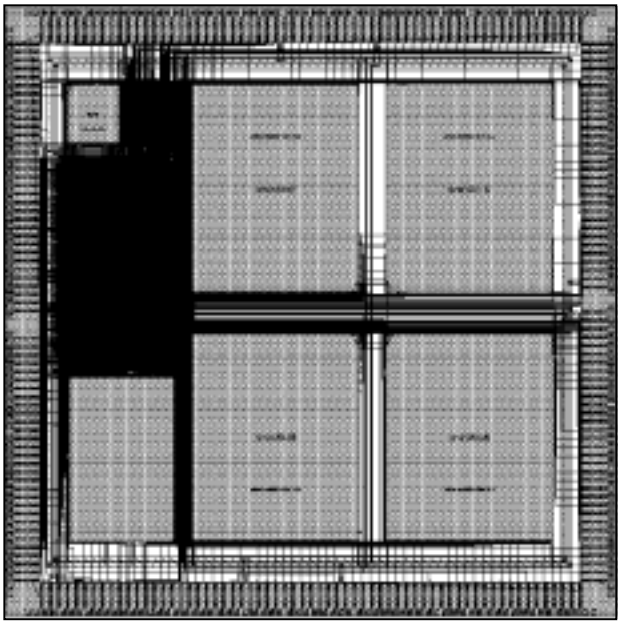

Figure 6: µPP circuit Layout

8. References

[1] *TOMI (Tools for OMI) Technical Annex*, Esprit Project 20724. Sept 1995.

[2] *TMPLUS System Hardware Description (Spanish)*. Telefónica I+D Internal Document. Oct 1997.

[3] *TOMI Deliverable 4.1.A: Circuit Specification*. May 1997.

[4] *AMBA. Advanced Microcontroller Bus Architecture Specification.* ARM IHI 0001D. April 1997. Advanced Risc Machines Ltd.

[5] Reference Peripheral Specification. ARM DDI 0062D. May 1998. Advanced Risc Machines Ltd.

[6] *Reusing VHDL Soft-Cores by Means of Using Appropriate Workspace Managment and Navigation Tools*. S. Olcoz et al. First International Forum on Design Languages. September 1998. Lausanne, Switzerland.

[7] *ARM Software Development Kit User Guide v 2.11.* ARM DUI 0040C. May 1997. Advanced Risc Machines Ltd.