A Programming and Functioning Learning Tool of a Digital Signal Processor

F.J. González Cañete[*,1], **S.J. de Hoyos Romero**

[1] UNIVERSIDAD DE MÁLAGA ETSI Telecomunicación, Dpto. Tecnología Electrónica, Campus de Teatinos, 29071 Málaga, España

This paper describes a tool developed for an easy learning of the programming and functioning of the TMS320C30 digital signal processor. This tool allows the student to develop and debug software for the processor at home simulating the real processor. It also includes additional functionalities not supported by the TMS320C30 software that allows a more accurate representation of data, registers and memory.

Keywords learning tool;programming;Digital Signal Processor

1. Motivation

The engineer career students usually have to learn the functioning of systems, instruments or machines that are not normally available in the market. This is the case of the Digital Signal Processors (DSP), that are microprocessors specifically designed for the processing of signals such as images and sounds. In that way, the students can not practice with the DSPs the time they consider necessary or in their free time because those systems are expensive and they are only placed in the laboratories of the University that are opened only at certain hours of the day. Another limitation is the number of systems available for the students and this is in fact the most important problem because the students have to share the equipments and systems.

These limitations made us decide to implement a software tool that simulates the functioning of the DSP based system studied in the subject "Advanced Digital Systems Laboratory" (*Laboratorio de Sistemas Digitales* Avanzados) in the speciality of Electronic Systems (*Sistemas* Electrónicos) of the Telecommunication Engineer career (*Ingeniero Técnico de Telecomunicación*) [1]. In this subject the functioning and programming of the Texas Instruments [2] digital signal processor TMS320C30 is studied and also some specific instrumentation that the students do not have at home such as oscilloscopes and signal generators is used.

2. The TMS320C30 and TMS320C30 EVM

The TMS320C30 [3] is a 32 bit floating point digital signal processor with Harvard architecture and the following characteristics:

- Eight general purpose 40 bit registers and eight 32 bit direction registers.
- Two blocks of 1K word of RAM and 4K word of ROM.
- A 64 word cache memory.
- Two serial ports for the communication with other systems.
- Two 32 bit programmable timers.
- An integer and floating point multiplier.
- An arithmetic logic unit (ALU).
- A 32 bit barrel shift.
- An auxiliary register arithmetic unit (ARAU).

[*] e-mail: fgv@uma.es, Phone: +34 9527176

The TMS320C30 has five memory addressing modes: the classical register, direct and indirect addressing modes and two specific ones, the circular addressing mode that is optimized for convolution operations and the bit reverse addressing mode that is used for the Fourier Transform operation.
The architecture of the DSP allows it to run two parallel instructions such as a multiplication and an addition, getting in that way the best performance of the microprocessor.

The TMS320C30 EVM is an evaluation module that includes a TMS320C30 DSP. This evaluation module is connected to a PC using an ISA port. It is used to develop, transfer and debug the source code developed in the PC. It also includes some peripherals for the TMS320C30. The architecture of the TMS320C30 EVM is shown in figure 1.

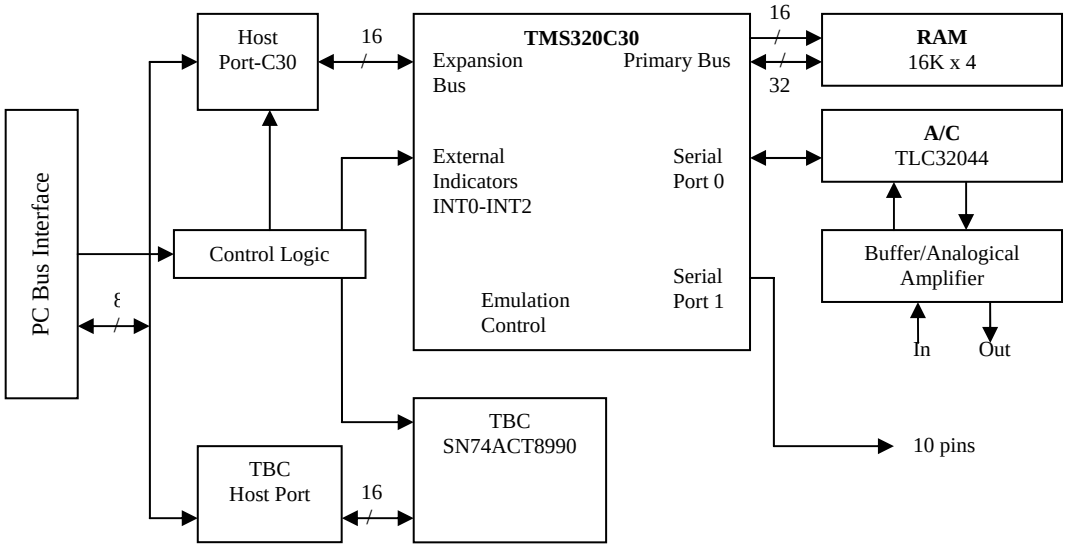

Fig. 1 TMS320C30 EVM architecture.

The TMS320C30 EVM contains:

- A TMS320C30
- 16K word of RAM
- An analog to digital and digital to analog (AD/DA) converter.
- An external serial port.
- An interface with the PC.

The AD/DA converter has a 14 bit precision and it is connected to two external RCA connectors, one of them for input signals and the other for output signal. This AD/DA converter is configured to accept and transfer mono sound signals.

The TMS320C30 EVM box also contains an assembler and C compiler and a linker to generate executable files. The executable files are loaded in a software tool called 'debugger' that transfers them to the TMS320C30 EVM using the ISA port of the PC. This tool allows debugging the programs and watching the register and memory values as well as modifying them, but it has the lack that it needs the TMS320C30 EVM board to work.

3. The environment

The main objective of the developed tool is to be an alternative to the 'debugger' simulating the functioning of the TMS320C30 EVM and adding some features that are not present in the 'debugger'. In that

way, the students can use the simulator at home and develop the exercises of the subject. Once the exercises have been developed and simulated, they can be executed in the real TMS320C30 to test the functioning in the real system.

This tool has been developed using the Borland C++ Builder 6 compiler and the executable program is oriented to the Microsoft Windows platform. The main window of the tool is shown in figure 2.

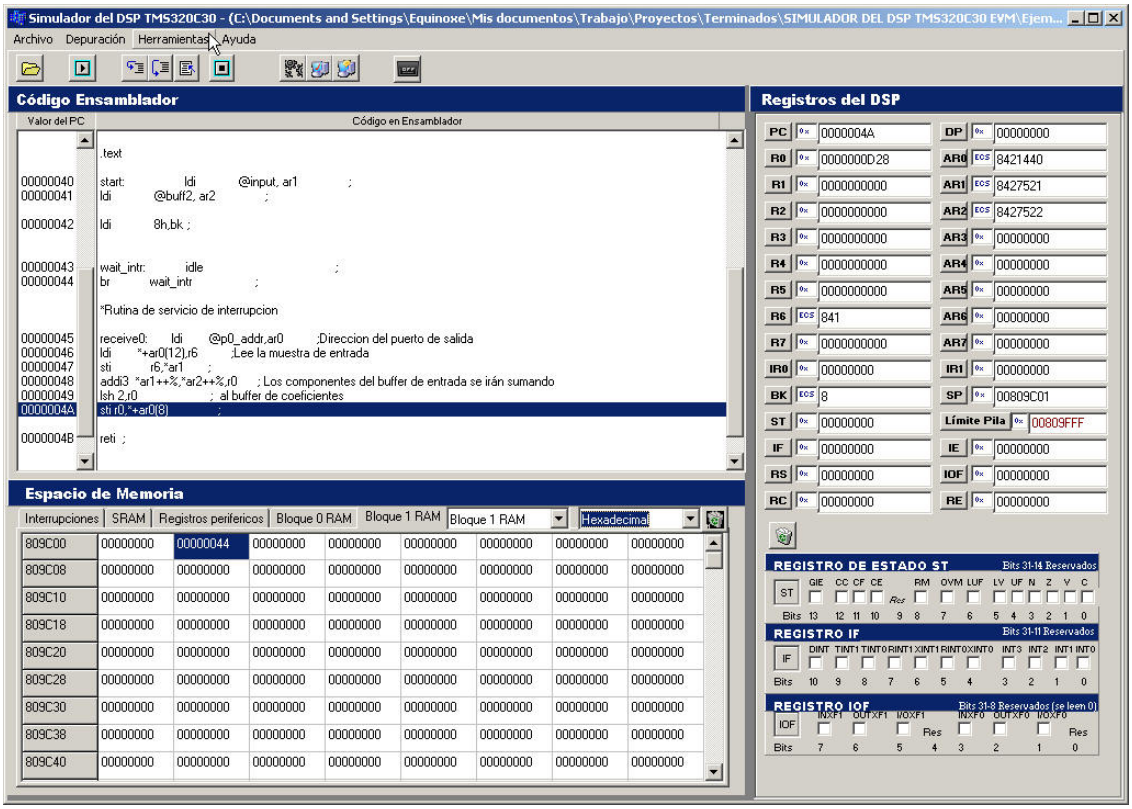

Fig. 2 Main window of the tool.

The main window of the tool is divided into four zones. The first of them is situated in the upper section and it contains the classical menus for accessing the options of the program and a tool bar used as a short access to the most important features of the program. The rest of the sections are explained in more detail in the next subsections.

3.1 The assembler code section

This section shows the source code loaded in the tool that is going to be simulated. This tool accepts only assembler source code of the TMS320C30, hence C source code is not accepted. We decided to implement only the assembler support because the C code is not used in the subject.

In the left of each assembler line the memory address where it is located is shown. The tool has the constraint that the source code has to be syntax error free if we want it to work properly. This is easy to get if we use the compiler program included by the TMS320C30 EVM that does not need the original board and hence the student can use it at home.

3.2. The Registers section

This section is located in the right side of the window and it shows the value of every register of the TMS320C30 DSP. In the upper side, a numeric representation of the auxiliary registers (AR0-AR7), the general purpose registers (R0-R7) and other more specific registers such as the PC (Program Counter), ST (STate) and SP (Stack Pointer) are shown. The format representation of the registers values can be modified to obtain a more human like or machine like view of them. In that way, the register values can be shown in hexadecimal, binary, signed integer, signed integer with scientific notation and floating point format. This is a great advantage with respect to the original 'debugger' that only shows the data using the hexadecimal format that is not a good human like representation. The value of the registers can be modified at simulation time if it should be necessary.

In the bottom of the section an alternative bit like representation of the ST, IF (Interruption Flags) and IOF (Input Output Flags) registers is presented. Each bit of these registers represents a special notification to the processor, consequently this representation lets us view and modify each bit of the registers.

3.3. The memory section

This section is situated at the bottom of the window and is divided into five tabs that represent the five memory sections of the TMS320C30 EVM memory map. These memory sections are:

- Interrupciones (Interrupts): It contains the pointers to the interruption functions.
- SRAM: Corresponds to the 16K words of external memory.
- Registros de periféricos (peripheral registers): It contains the registers mapped into memory to configure the peripherals connected to the TMS320C30 EVM such as the AC/DC converter.
- Bloque 0 RAM (RAM 0 block): It is the first block of 1K word of internal RAM memory.
- Bloque 1 RAM (RAM 1 block): It corresponds to the second block of internal RAM memory.

The memory positions are shown in a table mode representation with the physical address at the left of each line of eight memory positions. The data of the memory positions can be represented, as well as the registers, with five different representations: hexadecimal, binary, signed integer, signed integer with scientific notation and float and they can also be modified.

4. Working with the tool

The process to use the tool can be summarized into five steps:

1. An input file with 14 bit samples must be generated to be used as an input for the process. This file can be generated using a mathematical tool such as Matlab [4].
2. The source code is loaded into the simulation tool specifying the input file created in the step 1 and an output file that will contain the output of the simulation process. In this step, the sampling rate of the AD/DA converter must be selected. This feature is easier than the assembly hard coded mode needed to configure the AD/DA converter in the original TMS320C30 EVM.
3. The memory map of the simulation has to be selected. In the TMS320C30 EVM this is done in a special hard coded file that specifies the memory map to the linker. In this tool, this is done in a visual way using a typical configuration as default.
4. The sections of code must be assigned to the sections of memory. As well as the precious step, this is done in the TMS320C30 EVM with a hard coded file to the linker. The tool allows an easy way of assigning the sections.
5. Finally, the source code is simulated step by step or in a continuous run until the end of the input sampling file is reached.

The step by step run feature described in step 5 is the characteristic that allows controlling the execution of the assembler code. In that way, the students can observe the results of the execution of each source code instruction and how they modify the registers, the memory or the control of the program.

4. Conclusions

A tool that allows the student of the "Advanced Digital Systems Laboratory" subject in the speciality of Electronic Systems of the Telecommunication Engineer career to learn the programming and functioning of the TMS320C30 EVM digital signal processor has been developed. This tool is a Microsoft Windows based application that simulates the functioning of the original TMS320C30 EVM board with the advantage that the student does not need to have the board and the expensive instrumentation necessary to develop the exercises proposed in the subject.

References

[1] http://www.uma.es/ordenac/showProgDoc.php?cursoAcad=2006&TitCen=62&asigUMA=4388
[2] http://www.ti.com
[3] TMS320C30 User's Guide, Texas Instruments, Dallas, Texas, USA, 1988.
[4] http://www.mathworks.com