MARS - The X20 device and the SM1000 board

S. Cavalieri(*), G. Di Giugno, E. Guarino

I R I S S.r.l.
Parco La Serra 151 - 00018 Paliano (FR) ITALY
Fax. (775) 533343 - Tel. (775) 533441 - Email: nicola.larosa@euromik.it

(*) Department of Physics of the University of Napoli (Italy)
EMAIL CAVALIERI@NAPOLIINFN.IT

ABSTRACT

The X20 is an ASIC (Application Specific Integrated Circuit) conceived and designed in IRIS. It was intended to be used for research developments in new synthesis algorithms and architectures. The X20 is a fully programmable DSP with parallel operators, high interconnection freedom and addressing capability. It includes some useful facilities for musical applications. Two X20s with a general purpose controller (Motorola MC68302), their respective memories and glue logic useful for connection between them, lay on the SM1000 board, representing the core of MARS workstation. The SM1000 communicates by a MIDI port, an RS232 interface, a parallel port, a serial IIS standard port with the external world.

1. INTRODUCTION

The X20 was conceived to be a general purpose sound generator useful for further developments of the research both in the field of new synthesis algorithms and in that of musical instruments ASIC architectures. A programmable, real time and modular Digital Signal Processor seemed the best way to achieve a suitable tool to meet these needs (Allen 1985). So, continuing the experience of the synthesizers designed for over 15 years by members of the group (Cavalieri 1976, Asta 1980, Guarino 1983) with the recently acquired capacity to integrate circuits, the X20 was conceived and realised, we wanted to keep all the good peculiarities of the previous architectures such as high calculation speed, parallel and pipelined structure of the operators, low level programmability, while increasing its interconnection flexibility.

The modularity of the device was another requirement important to us: actually more than one X20 may work together; one of them being the master that synchronises all the other slaves. An easy communication among devices has been assured: both serial and parallel buses are available for that, the choice being suggested by the required data rate.

Conditioning of the dynamic flow, which is necessary for many non-linear operators, is also provided by a programmable conditional choice of ALU operands (cfr. 2.9). This solution easily guarantees a constant number of executed instructions, i.e. a constant sampling rate.

An easy way to accomplish most of the circuit test was simply providing an output to the Chaff (fig.1). Some more ad hoc solutions completed the test purpose from the point of view of the circuit. It was harder to solve the problem of verification and revision of arithmetic-logic results, as well as the production of the test vectors. The former was solved by development of a fast simulator written in C language. The latter needed the creation of a C-like language including proper macros oriented to the stimuli handling: this language was called SLANG (Vorhis 1990) and uses the C compiler to translate the source program into a form acceptable to the circuitual simulator. These tools were developed keeping in mind the goal of a high level architectural characterization, verification and synthesis (Seguin 1987, Camposano 1989).

The X20 was designed using the 1st Nand Standard Cell TOSHIBA library. Its instruction cycle minimum time is about 40 ns, which yields a sampling rate of 25 Khz through an execution of 512 instructions per sampling period. We received the first samples of X20 in June 1990, packed in a 144 pin square flat package, immediately checked them out and went on to design an application board in collaboration with ITIS called SM1000 (ITIS 1991) and described later in this paper.

2. THE X20 CHIP

The X20 has 4 communication paths with the external world:

- Serial I/O at programmable clock speed for direct connections to AD/DA converters or other X20s;
- parallel I/O including address, data and control signals to any kind of mass memory or to other X20s;
Regarding the architecture (see Fig.1), the X20 uses a 24 bit 2's complement fixed point (left normalized) representation of data. Two data memories (DMA, DMB) operate independently. The DMB has twice the dimension of DMA and some more addressing capabilities (cfr. 2.2). This structure is more expensive in terms of \( \mu \)P width, but has the advantage of gaining many cycles when the algorithms are implemented.

Data memories feed the 2 main operators (ALU and MUL) as well as other blocks devoted to control and auxiliary functions: Internal Tables & Constants (ITC), Table Address Generator (TAG), Control Registers (CTRL-REGS); crossed connections have been provided between ALU and MUL, while data coming from external tables is sent to MUL in order to produce linear interpolation. The interface (ITF) rules \( \mu \)C access requests. A slow rate calculation of some portion of the same instruction code on 64 different sets of parameters is possible. Each iteration of this portion of code can be executed at an individually programmable rate chosen among 1/64, 1/128, 1/256 or 1/512 of the full sampling rate.

The chip has a 16 bit static configuration word (CPFW), and two dynamic control words dedicated to individual oscillator table choice (CWO, cfr.2.2) and to individual presets for reduced rate operations, mainly envelope calculation (CWE). The static one is directly written by the \( \mu \)C and contains information not to be changed during normal functionality, such as resets and device status flags: presets concern write enable flags, DRAM size selection and serial output clock period selection; status is defined by interrupt enable and flags, slave/master status flag, device on/off flag.

The dynamic words are loaded under \( \mu \)P control from DMB as often as needed during the execution of the \( \mu \)P itself. The two interrupt lines are respectively dedicated to envelope calculation and conditionally met events (zero-crossing, comparison between variables, etc.). Options for both lines are loaded into CWO and give one the freedom to choose, for each channel, a DMB auxiliary table size (cfr. 2.2) and a reduced sampling rate for envelopes.

2.1 \( \mu \)C Interface

The X20 chip has to be fed by a \( \mu \)C which presets the whole chip memory space, updates the values of the parameters involved in algorithm execution during run-time, fills the \( \mu \)P RAM with the proper code and handles interrupts issued by the device when programmed conditions are met. All these targets are mapped in 8 kbytes of the memory space of the \( \mu \)C itself, so the whole set of memory instructions can be used, programming is straightforward and data transfer is faster. The 8 bit width of the data bus allows interfacing to any kind of \( \mu \)C, as long as the speed of access satisfies the required data rate.

It is obvious that, when the X20 is off, the whole addressable space is immediately reached by the \( \mu \)C without any handshaking protocol being needed. Something must change when the X20 status is turned on: only the CPFW is still accessed in immediate mode. Access to \( \mu \)P has to be inhibited because the X20 needs reading instructions every cycle.

Figure 1. X20 architecture

349
Finally, data memories can be accessed in accordance with a periodic time window provided by the µP. During each of these time windows internal DM operations are disabled while eventual µP access requests are satisfied and the handshake signal, that was keeping the µP in a wait state following its request, is released. So the latency time depends on the frequency of the time windows; this also means that the maximum transfer delay is user programmable. Using preprogrammed time windows as described avoids altering the steady flow of operation required by the fixed sampling rate.

2.2 Data memories and Table Address Generator (TAG)

The µP provides several fields describing at each cycle the addressing modes and data format rules to access data memories. Two absolute addresses (8 bits and 9 bits respectively) are available for DMA and DMB. They may be combined with slot-counter for a time-dependent displacement that allows reduced rate treatments on multiple sets of parameters. DMB can also be addressed into auxiliary tables (whose size is chosen after a CWE load) containing a portion of the ALU output Chex (indexed addressing) or through an autocycle which registers (circular addressing). The available data formats are: LONG (24 bits), WORD (16 MSH), BYTE (8 bits), UNSIGNED BYTE (8 bits).

The TAG block generates the physical addresses for the table memories (both internal and external) receiving the CW0 and a 24 bit phase. It also produces the next sample address and the fractional part of the phase, to be used for the interpolation between samples. Physical address computation is performed by modifying the proper number of MSB of the phase in accordance with the CW0 information, in a way that preserves the physical meaning (frequency) of the table look up step. Therefore, the duration of a complete sweep in the table does not depend on the table length but only on the frequency value. The code contained in the 16 MSH of CW0 allows choosing a table size from 32k to 1 Mw by steps of 2¹ powers (free size tables are addressable by means of programmed address control), four more bits identify one of 16 pages, bringing the whole addressable space to 16 Mw. This applies no matter what kind of memory (SRAM, DRAM, ROM) is attached, because the µP code allows programming of all the control signals.

2.3 ALU and conditional logic

Both input operands (Ain, Bin) come from selection by a pair of registers. They can be added or subtracted, while the following operations to a condition: if the result is negative: overflow and underflow detection and correction; positive sign protection for unsigned variables.

As it's been told, we preferred using conditional operations over conditional branches in order to modify dynamic flow. The number of cycles needed to perform a conditional choice would be almost the same in both cases, but some extra logic ought to be added to handle conditional branches and the static flow would have exceeded 512 instructions.

Let us now analyse the way the conditioning is actually made. The signs of Bin operand (SB) and of the uncorrected output (SC) as well as the carry-out (CY) can be registered under µP control. Conditional logic is based upon comparison between two of those flags. As the flags are also presentable to 0 or 1, comparisons with a reference value are also possible. If we identify equality as a true condition and inequality as a false one, then the following choices are available:

choice 1: true ↔ (0, Bin) / false ↔ (Ain, 0)

choice 2: true ↔ (0, Bin) / false ↔ (Ain, 4D0)

In other words, choice 1 enables one of the operands; choice 2 enables or disables operand Ain.

3. THE SM1000 BOARD

The core of the MARS workstaton is the SM1000 sound generation board (see §5.2); two X205 are connected in a master/slave configuration to produce sound samples. This DSP pair is controlled by a general purpose µC (Motorola MC68302) with its program and data memory. Most of the glue logic is implemented on PLAs. Surface mounting technology is employed to reduce the board area. The board has 6 512kbit layers.

The SM1000 is attached on one side to a low computer, on the other side to a DAC/ADC board called ADDA (ITT 1991), and has the following external ports: a MIDI port, an RS232 interface and a parallel 16 bit expansion port. The 16 MHz MC68302 has the task of interfacing the sound generators to the external world, in order to properly configure the X205, control the real time sound generation, and also switch to one of four available µP blocks.

The µC handles interrupt requests from the X205, the RS232 serial link and the parallel port, as well as MIDI messages. The board is organized around the 68000 bus to which are connected: the µC, the two X205, the memory for the µC composed of a DRAM block (256 Kw) and an EPROM block (128 Kw) both expandable to 1 Mw and containing the real-time proprieties operating system RT20M (Andrenacci 1992). In addition to the normal mode of operation an emulation mode is allowed, using two connectors which deliver the bus of the µC.
The X20n runs at 40 MHz and produces audio samples at 39.0625 KHz (a higher sampling rate can be obtained reducing the number of instructions per period). They are connected through their serial interface for communication from the master to the slave. Each chip has its own 16 K-byte RAM containing four blocks of 512 instructions (total 2Kx64). Each X20 has its own port to a bank of memory laying on a small added board called DREP (ITIS 1991). This has been separated in order to easily configure amount and type of memory up to 32 Kbyte per X20. The actual configuration on the SM1000 provides 4 Kwords of DRAM (100 seconds of sounds or delays) and 256 Kwords of PROM (6 seconds of preset sounds). The MIDI port is provided for communication with MIDI world. The RS232 port was used during the debugging phase of operating system development for downloading programs to the µC, thanks to an EPROM Monitor/Debugger on board. The parallel bidirectional port connects the µC to the host as a data transfer rate of 800 Kbytes/sec. mainly with the purpose of downloading data to the table memories, to interface the board to external mass storage and to allow higher level graphical and sequencing control (Armani 1992). In the actual MARS configuration the host is an ATARI ST, connected via VME bus, on which more than one SM1000 board could be connected.

The serial port to ADDA board, working at 5 Mbits/sec, provides connection to 8 DAC and 4 ADC channels. In detail ADCs are connected to the X20 master, the X20 slave to the slave, both master and slave to DACs.

Figure 2. SM1000 board schematic

ACKNOWLEDGMENTS

Special thanks to Jean Luc Pavy of ITIS, realiser of the SM1000, and to Renato Bessegato of IRIS who took care of the test board and designed the parallel interface.

REFERENCES

- E. Guarino, G. Di Giugno "Un processore rapido floating point" Atti del CIM, 1983, Ancona (Italy)
- R. Campana, W. Rosenstiel "Synthesizing circuits from behavioral descriptions" IEEE transactions on CAD, vol. 8, N2, February 1989
- M. Vorhat, A. Prestigiacomo "SLANG (Simulazione linguaggio manuale) IRIS internal document, 1990
- ITIS "SM1000, DREP and ADDA technical description" 1992