Philips Extends 8051 Architecture to 16 Bits

New Instruction Set Provides Easy Transition from Venerable 8051

by Linley Gwennap

Offering increased performance to current 8051 customers, Philips Semiconductors (formerly Signetics) has announced a 16-bit instruction set called the 8051 XA, for extended architecture. A new CPU core implementing the XA design offers four times the performance of the 80C51 at the same clock speed, with larger gains on programs that use 16-bit data. The new architecture also fixes some problems in the original 8051 design, which dates back to the late 1970s, to improve performance and better support compiled languages.

Philips plans to deploy a range of products using the new core, as it has done with the 80C51. These products will include various combinations of on-chip memory and peripherals. For now, the company has announced a single part, known as the XA-1, that is nearly function- and pin-compatible with the 80C51. Although the company has not yet received first silicon, it hopes to sample the XA-1 by the end of this year, with volume production in 2Q95.

Instead of leaping directly to 32 bits, as Zilog has done with its Z380 extension *(see **080901.PDF**)*, Philips chose to walk the fine line of price/performance in the no-man's land of 16-bit microcontrollers. The company has many customers that need more performance than the 8-bit 80C51 can provide but aren't willing to pay $10 or more for 32-bit performance. Using a 16-bit core, the company plans to sell the XA-1 for $7, just a few dollars more than an 8-bit part, and to eventually bring the price below $5 through die shrinks and design improvements.

Code- But Not Binary-Compatible

To meet customer demands for increased performance, Philips could have turned to an existing 16-bit or 32-bit architecture, perhaps even a RISC architecture. Many of its customers, however, have invested dozens of engineer-years in developing 8051 code; some no longer even have the source code for their programs. Furthermore, these engineers are comfortable and skilled with the 8051 instruction set. Thus, Philips decided to retain as much compatibility as possible with the existing 8051 architecture, which sold more than 120 million units in 1993 alone.

Unfortunately, the original 8051 architects left little room for expanding the instruction set. In particular, there are no spare encodings in the first instruction byte. Retaining full binary compatibility would have required making most of the new instructions three or more bytes in length, an undesirable performance penalty. Two-byte instructions are ideal in a 16-bit machine because they fit the bus width, so Philips decided to re-encode the 8051 instructions to make room for the new extensions. As Figure 1 shows, 43% of the 479 instructions—and most of the frequently used ones—are encoded in one or two bytes in the XA instruction set.

The XA assembler supports all 8051 mnemonics, so customers with source code can run it through the new assembler without modification and generate XA binaries. Because there is a one-to-one mapping from 8051 to XA instructions, there should be few (if any) performance issues when moving to the new architecture. Furthermore, programmers familiar with 8051 assembly language can easily program the new processor.

The drawback to the Philips approach is that old binaries will not run on the new processor. Because all 8051 instructions are present in the new instruction set, it is easy to translate old binaries to new binaries, and Philips provides a translator to do just that. Unfortunately, the length of some 8051 instructions was increased to make opcode space for the new XA instructions, so the translated binary will expand slightly, changing the instruction addresses. The translator handles simple cases, like branches, but some sequences—jump tables, for example—must be converted by hand.

A few other compatibility problems may crop up. Like most processors, but unlike the 8051, the XA grows its stack down from the top of the memory space. Because the cycle count of most instructions has been reduced from 12 to 3, some timing loops may need to be rewritten. But on the whole, the Philips approach should allow most existing 8051 programs to migrate to the new processor with little or no hand coding.

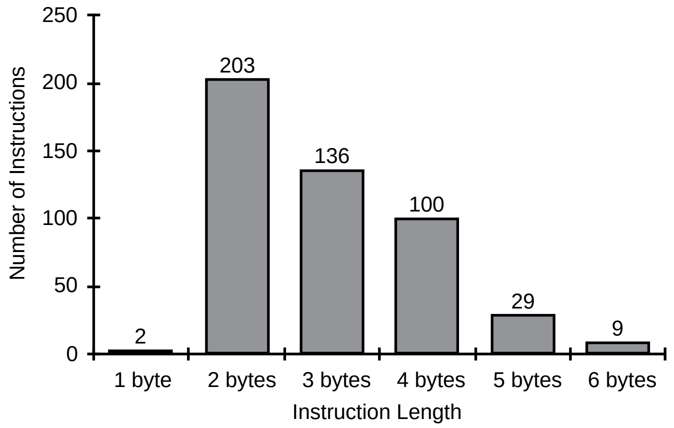

Figure 1. Of the 479 instructions in the 8051 XA instruction set, 43% use a two-byte encoding and only 8% use more than four bytes.

Group 1: Arithmetic	Group 2: XCHG	Group 5: Jump, Call
Rd, Rs	Rd, Rs	rel16
Rd, [Rs]	Rd, [Rs]	[Rs]
[Rd], Rs	[Rd], Rs	addr24†
Rd, [Rs+offset8/16]	Rd, direct	[A+DPTR]‡
[Rd+offset8/16], Rs		
Rd, [Rs+]	**Group 3: MOV**	**Group 6: CJNE**
[Rd+], Rs	Group 1 plus:	Rd, direct
direct, Rs	[Rd+], [Rs+]	Rd, #data8/16
Rd, direct	direct, [Rs]	[Rd], #data8/16
Rd, #data8/16	[Rd], direct	
[Rd], #data8/16	direct, direct	**Group 7: Shift**
[Rd+], #data8/16	Rd, USP	Rd, Rs
[Rd+offset8/16], #data8/16	USP, Rs	Rd, #data4
direct, #data8/16	**Group 4: MOVC**	
Rd, #data4*	Rd, [Rs+]	**Group 8: MUL, DIV**
[Rd], #data4*	[Rd+], Rs	Rd, Rs
[Rd+], #data4*	A, [A+DPTR]	Rd, #data8/16
[Rd+offset8/16], #data4*	A, [A+PC]	
direct, #data4*		

Table 1. Philips' 8051 XA architecture provides a much wider variety of addressing modes than the original 8051 instruction set. *ADD, MOV only. †FJMP, FCALL only. ‡JMP only.

Many Improvements to 8051

Philips' first task was to expand the register width to 16 bits. The separate code and data spaces become 16M each. The instructions use a linear code space, but the data space is broken into 64K segments, providing better backward compatibility as well as some protection in a multitasking system.

The new architecture adds more registers for a total of 16 general-purpose registers, each 8 bits wide. The design breaks the accumulator bottleneck by allowing generalized register-to-register operations. Any register pair can be used for addressing, avoiding another 8051 bottleneck: the single data pointer. Table 1 lists the addressing modes available for various instructions. Like the 8051, the XA architecture has four banks of registers, providing fast (2 µs) context switches.

As Table 1 shows, the arithmetic, logical, and move instructions support direct, indirect, and indirect+offset addressing modes. These instructions can operate on 16-bit quantities by addressing a register pair; 16-bit operations use the .W extension—for example, MOV.W or ADD.W. The .B extension is used to access individual 8-bit registers; for compatibility with 8051 assembly code, this mode is the default.

Taking advantage of the new opcode arrangement, Philips created a short (4-bit) immediate addressing mode for the ADD and MOV instructions. These "short" instructions (ADD.S and MOV.S) take two bytes to encode, instead of three or four bytes for a longer immediate value. The ADD.S instruction eliminates the need for increment and decrement instructions. Another improvement, auto-increment addressing, simplifies loop constructs.

Philips beefed up the XA instruction set with new integer multiply and divide instructions. These accept 8- and 16-bit operands (or 32 bits for divide) and calculate up to 32 bits of result. Both signed and unsigned options are available. New branch instructions expand the number of conditions supported for signed and unsigned comparisons. New FJMP and FCALL instructions use a 24-bit address to directly access the entire code space. Table 2 shows the entire XA instruction set.

The new architecture includes some features to simplify multitasking operating systems. Two stack pointers are used, one for the OS and one for the current user task. This prevents a task from destroying the OS stack. Each task can be assigned its own 64K data segment. Intersegment fetches can be disabled for each task. Interrupts and exceptions use the system stack pointer.

XA-1 Offers Three-Cycle Execution

The XA-1 processor is the first implementation of the 8051 XA architecture. It executes register-to-register arithmetic instructions in three cycles, four times faster than the standard 80C51. Most MOV instructions, even those that use indirect addressing, also execute in three cycles. Use of the auto-increment mode adds an extra cycle, while the indirect+offset mode costs two additional cycles. Most jumps take five or six cycles and calls typically take seven. Conditional branches require three cycles if not taken, six if taken.

The XA-1 includes a 16×16 multiplier in hardware, which accelerates integer multiply and divide operations. The MUL16 instruction completes in 12 cycles (480 ns at 25 MHz), while DIV32 takes 22 cycles.

Figure 2 shows a block diagram of the XA-1. In addition to the XA core, the chip includes 32K of ROM and 512 bytes of RAM. The on-chip memory is 16 bits wide and is fast enough to keep up with the three-cycle peak instruction rate.

The XA-1 also includes a basic set of on-chip peripherals: four 16-bit counter/timers, two UARTs, and four 8-bit I/O ports. Additional memory and I/O devices

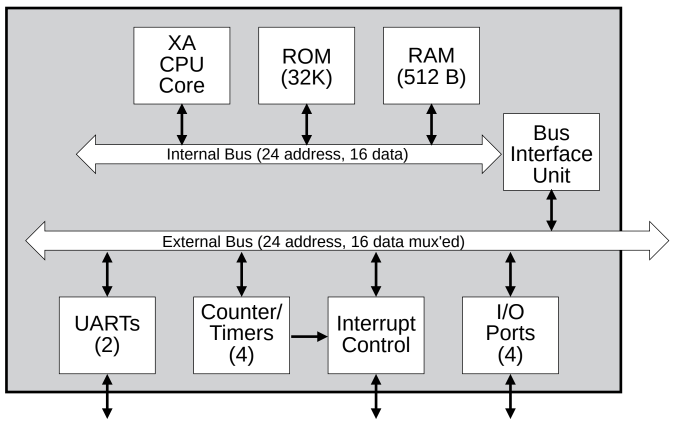

Figure 2. Block diagram of the XA-1, which includes a large on-chip ROM plus a small RAM and some basic peripherals.

Price & Availability

Philips expects to sample the XA-1 in December with volume production in 2Q95. The mask-ROM version will sell for $7.00 in 10,000-unit quantities.

Contact Philips Semiconductors (Sunnyvale, Calif.) at 408.991.5207; fax 408.991.3773.

can be connected via the external bus, which multiplexes 24 bits of address and 16 bits of data. The bus can be configured to operate with only 8 data bits and as few as 12 address bits, decreasing the cost of connecting I/O devices. Bus signals that are not used can be configured as extra parallel I/O pins.

With a 5-V supply, the processor is expected to run at 25 to 30 MHz, or nearly 10 peak native MIPS. Philips believes the XA-1 will reach 20 MHz at 3 V. The company has not verified this performance, as no silicon has been produced. The initial chip uses a synthesized logic design and will be built in a 0.8-micron CMOS process. Philips would not reveal the core die area, but with a synthesized design, this figure would be fairly meaningless: by the time most XA-1 designs reach volume production, Philips expects to have available a much smaller hand-packed core in a half-micron process.

Philips estimates that the chip will draw 350 mW at 30 MHz but expects this value to drop dramatically in the new process. The XA-1 uses the same 44-pin package and pinout as the 80C51, except that two previous no-connect pins are now used as extra power and ground pins to support the higher power dissipation. The RESET signal has been inverted (it is now active-low), preventing the new chip from working in an 80C51 socket; the redesign, however, should be trivial in most cases.

Breaking Away from the 8051 Pack

With 10 vendors now marketing 8051-compatible processors, Philips needs to differentiate itself from the pack. The XA-1 provides a unique performance point, allowing current 80C51 users to migrate easily to a faster CPU with little hardware or software redesign.

Its toughest competitor may be another enhanced 8051, Dallas Semiconductor's 80C320 *(see **0705MSB.PDF**)*. The '320 has a basic instruction time of four cycles and adds some new architectural features, such as a second data pointer. It is fully pin- and binary-compatible with the 80C51 and sells for $6.50.

The XA-1 is 33% faster on basic instructions than the '320 and offers more significant architecture enhancements, including 16-bit computation. For those 80C51 designers willing to make the small extra effort to compensate for the differences in the XA-1, it should deliver significantly better performance. ♦

Instruction	Addr Modes	Description
Arithmetic, Logical, and Move		
ADD, ADDC	Group 1	Add (with carry)
SUB*, SUBB	Group 1	Subtract (with borrow)
CMP*	Group 1	Compare
AND, OR	Group 1	Logical AND, OR
XOR	Group 1	Logical exclusive OR
XCH	Group 2	Exchange values
MOV	Group 3	Move data
MOVC	Group 4	Move to/from code space
MOVX	Rd, [Rs] [Rd], Rs	Move to/from external memory
PUSH, PUSHU*, POP, POPU*	direct multiple	Push, pop from stack (or user stack)
Jumps, Call, and Returns		
JMP, FJMP*	Group 5	Jump (far jump)
CALL, FCALL*	Group 5	Subroutine call (far call)
RET	none	Return from subroutine
RETI	none	Return from interrupt
BCC, BCS	rel8	Branch if carry clear (set)
BEQ*, BNE*	rel8	Branch if equal (not equal)
BGT*, BG*	rel8	Branch if greater than (unsigned)
BLT*, BL*	rel8	Branch if less than (unsigned)
BGE*	rel8	Branch if greater than or equal
BLE*	rel8	Branch if less than or equal
BPL*, BMI*	rel8	Branch if plus (minus)
BOV*, BNV*	rel8	Branch if overflow (no overflow)
BR	rel8	Branch unconditional
JB, JNB	bit, rel8	Jump if bit set (not set)
JBC	bit, rel8	Jump if bit set and clear bit
JZ, JNZ	rel8	Jump if zero (not zero)
DJNZ	Rd, rel8 direct, rel8	Decrement and jump if not zero
CJNE	Group 6, rel8	Compare and jump if not equal
Shift, Rotate, and Bit Operations		
ASL*, ASR*	Group 7	Arithmetic shift left (right)
LSR*	Group 7	Logical shift right
RL, RR	Rd, #data4	Rotate left (right)
RLC, RRC	Rd, #data4	Rotate left (right) with carry
NORM*	Rd, Rs	Normalize
CLR, SETB	bit	Clear bit, set bit
ANL, ORL	C, bit C, /bit	AND (OR) logical with bit
MOV	C, bit bit, C	Move to/from bit
Miscellaneous Arithmetic, Logical, and Control		
MULU8	Group 8	Multiply 8 × 8 → 16 bits unsigned
MULU16*	Group 8	Multiply 16 × 16 → 32 bits unsign
MUL16*	Group 8	Multiply 16 × 16 → 32 bits signed
DIVU8	Group 8	Divide 8 / 8 → 8 unsigned
DIVU16*	Group 8	Divide 16 / 8 → 8 unsigned
DIV16*	Group 8	Divide 16 / 8 → 8 signed
DIVU32*	Group 8	Divide 32 / 16 → 16 unsigned
DIV32*	Group 8	Divide 32 / 16 → 16 signed
DA, SEXT*	Rd	Decimal adjust, sign extend
NEG*, CPL	Rd	Negate, complement
LEA*	Rd,Rs+Off	Load effective address
NOP	none	No operation (one byte)
BKPT*	none	Breakpoint
RESET*	none	Reset chip
TRAP*	#data4	System call

Table 2. The 8051 XA instruction set. Table 1 lists additional addressing modes. *Instructions added to base 8051 instruction set.