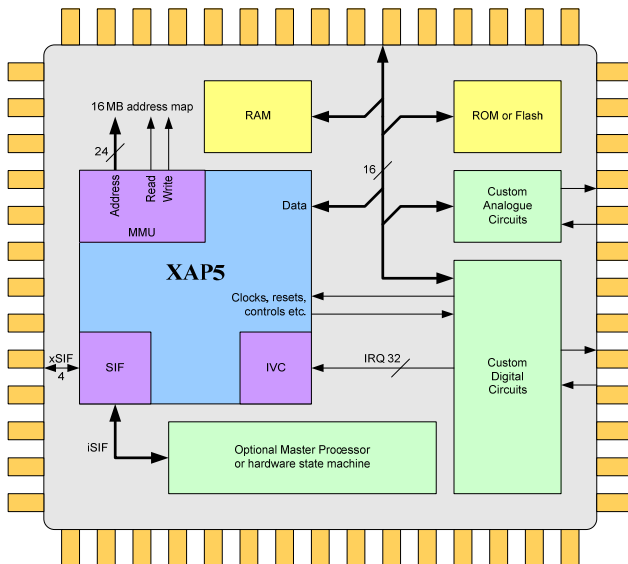


## XAP5 processor core



XAP5 is a 16/32-bit processor IP core offering advanced computing functions at the lowest possible cost and energy consumption.

XAP5 addresses the requirements of high volume and cost sensitive chip designs such as sensor nodes and wireless products, e.g. Bluetooth, ZigBee, GPS, RFID and NFC. Applications for XAP5 include automotive, communications, energy, medical devices, industrial, retail, metering, consumer and many other products.

When programs are stored in Flash memory and energy use must be minimised, as in battery-powered systems, then a XAP5 processor is the ideal choice.

Cambridge Consultants has developed and licensed the XAP<sup>®</sup> processor family since 1994. We also use these advanced energy-efficient processors in our ASIC design projects and they have now shipped in over a billion chips manufactured by our clients and licensees worldwide.

### XAP5a performance

- 16/32-bit, 16 MByte processor
- Eight general purpose 16-bit registers
- 24-bit registers: Program Counter, two Stack Pointers, Vector Pointer, four Breakpoints
- User mode and three privileged modes: Trusted, Supervisor, Interrupt
- 32 interrupts including four NMI
- 32 exception vectors
- Interrupt latency of 13 to 21 cycles
- Von Neumann architecture with little-endian organisation
- High code density – better than leading 32-bit processors
- Typical 65 nm routed area 0.037 mm<sup>2</sup>
- Core synthesis 18k gates (NAND2x1)
- 0.93 DMIPS/MHz from 16-bit memory
- Fast 205 DMIPS at 220 MHz\*
- Low power – 6 μW/MHz\*\*
- High efficiency – 155 DMIPS/mW
- xSIF serial debug interface
- xIDE with GNU GCC C compiler
- Soft core, Verilog RTL delivery

Near mode performance on 65 nm GP CMOS  
\* worst case 0.9 volts \*\*typical case 1.0 volts

### Low cost, low energy

XAP5 is a 16/32-bit processor core offering high performance and low energy consumption in a very small die area. Its modern architecture and software tools enable memory system design to be optimised for highly efficient systems on a chip.

XAP5 has a 16-bit data bus to memory and a 32-bit internal bus between registers and the ALU. Registers are automatically paired for 32-bit data or 24-bit addresses.

A processor with 16-bit data is the best choice for many applications, whereas 8-bit is insufficient and 32-bit is often unnecessary. Using 16-bits halves the size of a processor and its RAM compared to a 32-bit core. A 16-bit stack saves on size, cost and energy consumed by each access, e.g. on every push or pop.

XAP5 features very high density program code using a run-time mix of 16, 32 and some 48-bit instructions. Its short pipeline architecture maximises instruction throughput at low clock speeds, enabling many programs to run directly from Flash memory and consume less energy.

### Advanced computing

XAP5 brings advanced computing functions to low cost, low energy chip designs. Many of the functions support high reliability and security.

The processor runs programs using execution modes that isolate user tasks from privileged code such as the operating system and interrupt services. This protects critical code and data against any unintended modification or corruption and it can guarantee a real-time system's performance and responsiveness.

XAP5 programs can be stored anywhere in memory regardless of where they are linked to run from. Programs start up quickly and can execute in-place from non-volatile memory such as Flash. Multiple programs, or instances of the same program, can co-exist in memory facilitating reliable Flash updates and ease of software distribution.

Designers can organise their memory with custom MMU designs that secure XAP systems against hacking or reverse engineering. Attempting an illegal access is detected and causes an exception.

### Reduced memory cost

XAP5 has a regular, orthogonal, load-store instruction set with instructions available in 16, 32 or 48-bit forms depending upon the range of arguments. The linker automatically selects the shortest form it can use for each instruction. This carefully designed instruction set is optimised for XAP5's register layout and GNU GCC C compiler.

The resulting code density is as good as, if not better than, leading 32-bit cores running 16-bit program code. It is over twice as good as popular 8-bit cores. High code density reduces memory size and cost. It also takes fewer instruction fetches, thereby using less energy.

Programs are easy to write or port to XAP5 with its single flat memory map, unaligned data access and byte addressing. Applications can be written entirely from C without the need for assembly language.

XAP5 can address up to 16 MBytes of memory for programs and data. However, many applications only need a data memory of 64 kBytes or less in which case a C compiler option can run XAP5 in a Near data mode with a 16-bit address for data and 24-bit address for instructions.

Benchmark performance in Near mode improves by 30% and code size is approximately 10% smaller.

Access to data in all 16 MBytes is still possible from Near mode by using assembly language routines.

### Fast real-time performance

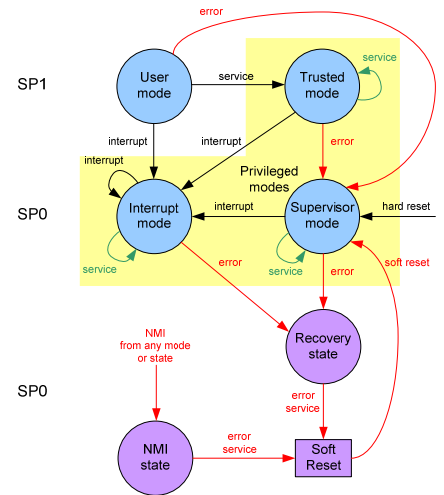
XAP5 switches rapidly between modes and tasks in response to interrupt or exception events that can be recursive to any depth. Switching is atomic with processor hardware storing and retrieving live data and state, to and from the stacks; SP0 and SP1. One stack persists for both Supervisor and Interrupt modes, while the other is used for User mode tasks and Trusted mode system calls.

Protection against misbehaving or malicious User mode software is provided by privileged instructions, i.e. they only execute in Trusted, Supervisor or Interrupt mode.

The result is a deterministic and minimised event response time with fast and secure task switching that maximises pre-emptive RTOS performance and reliability.

RTOS ports for XAP5 include CMX, FreeRTOS and Micrium uC/OS-II with others to follow.

XAP5's event handling provides a fast real-time interrupt response without any penalty for nested interrupts. Interrupt entry latency is between 13 and 21 clock cycles, depending on the time required to complete the current instruction. Longer instructions, e.g. multiply, divide and block copy can be interrupted. There are 28 maskable and four non-maskable interrupts. The interrupt controller hardware allows software assignment of 16 levels of interrupt priority.

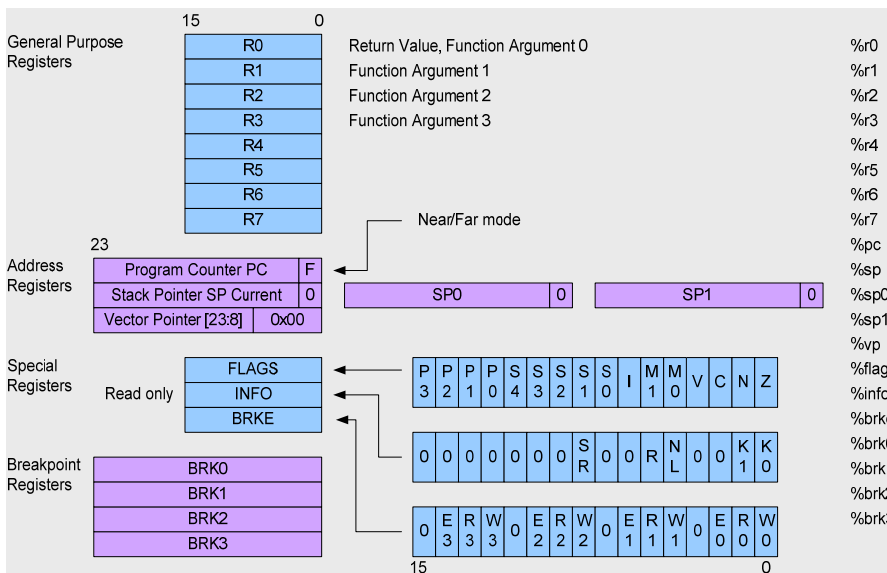


XAP execution modes

### Rich feature set

XAP5 incorporates a rich set of features that extend performance, functionality and flexibility:

- Multiply/divide. An internal unit accelerates multiplies and divides, performing a 16x16 multiply in just four clock cycles.
- Signal processing. Instructions that do a normalised multiply to adjust dynamic range.
- Endian swap. Instructions to rearrange bits or bytes between big and little-endian form.
- Single bit i/o. Set or clear selected bits at any address using atomic instructions.
- Custom Logic Unit. Customised instructions can be added to the processor, executed by custom hardware connected directly to the XAP core. A set of instruction templates, which preserve compiler performance, pass data directly to and from XAP5's general-purpose registers.
- Sleep mode. An instruction to stop the processor between tasks for minimum stand-by energy consumption. A wake-up signal restarts execution.
- iSIF internal interface. Enables XAP5 to act as a slave to an on-chip master processor or state machine. Used if XAP is part of an IP sub-system in a SoC or for saving registers prior to powering down the core to stop energy loss by leakage current.



### Flexible hardware

The standard XAP5 hardware is a single processor core including iSIF interface and xSIF debug, which is used unmodified in a chip design. A Memory Management Unit and an Interrupt Vector Controller are also provided and these are modified by the chip designer to achieve the required architecture, i.e. memory map with i/o registers and interrupt structure. XAP5 can be extended with an optional Custom Logic Unit.

XAP5 has a unified program and data bus structure with all vectors, instructions, data and i/o registers appearing in a single 16 MByte Von Neumann address map. This is preferred for low cost, low energy systems as the division of program memory into instructions and data remains entirely flexible. Typical XAP5 chip designs use only one 16-bit Flash and one 16-bit RAM.

XAP5's target applications are low cost and low energy systems that dictate the choice of Von Neumann over Harvard architecture. Harvard designs are complicated and less flexible as they must map or copy constants and initialised data from program memory to data memory before execution. At worst, this requires extra memory blocks in a chip design. Any speed advantage from Harvard is often lost in small chips, especially when executing from slow access Flash memory.

### Execute from Flash memory

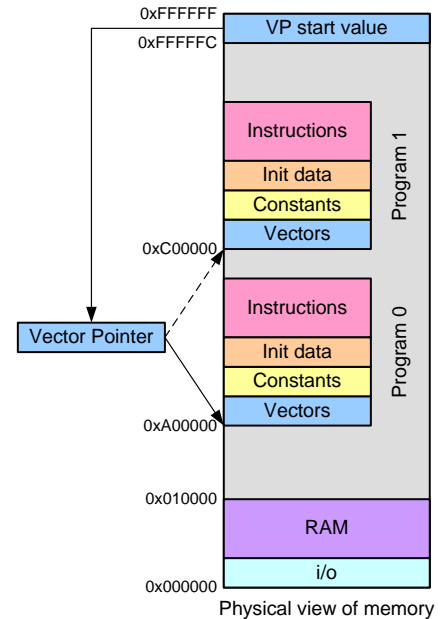
XAP5 is very efficient at executing programs stored in non-volatile memory such as OTP ROM or NOR Flash that is embedded on-chip or connected off-chip, perhaps in a System in Package (SiP).

Flash memory is large and offers a very low cost per bit, but has slow access time. XAP5's design gains optimum performance from Flash without compromising its capability to run fast from RAM or ROM.

XAP5's Vector Pointer holds the address where a program's event vectors are stored, including reset. In-place execution of programs can then be started from wherever they are located in memory. This means that programs start up immediately without having to first copy vectors or instructions to another memory.

Multiple instances of programs can co-exist in memory. This provides a safe mechanism for Flash program updates whereby a download can complete safely and be checked for errors before changing the VP start value and restarting the system.

XAP5 and its MMU also support an Address Translation Window that dynamically re-locates programs from physical storage to logical memory. This allows programs to be easily distributed and stored anywhere in memory, ready for immediate in-place execution.



XAP Vector Pointer

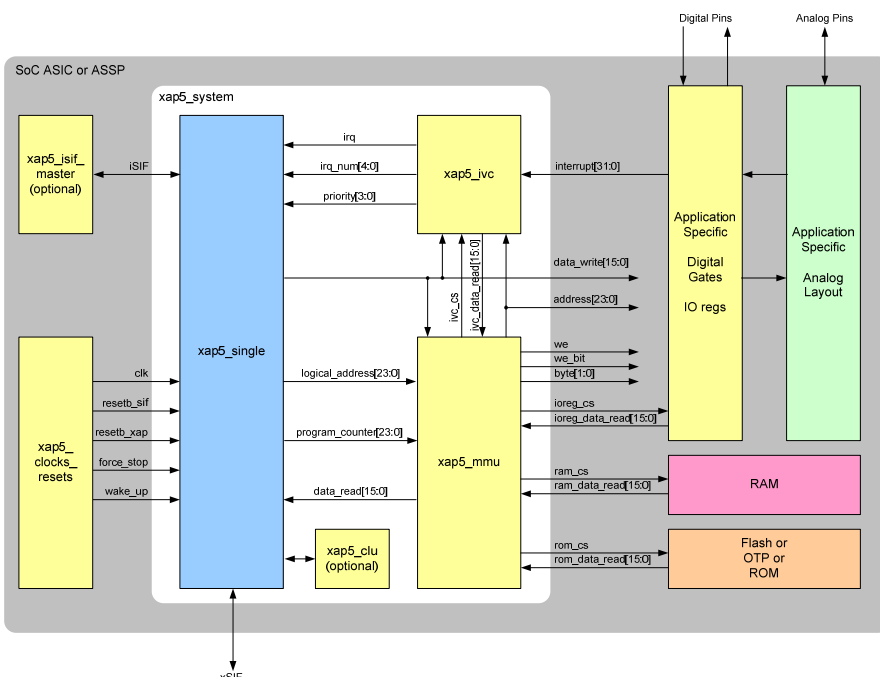
### Secure and reliable

High security systems, resilient to hacking or reverse engineering, can be designed with XAP5. The processor's software modes, event handling, deterministic behaviour and configurable MMU facilitate various techniques that support a secure software kernel managing isolated tasks in a trusted and high-availability system.

For example a section of code held in ROM may be used to access security keys stored in memory. By gating addresses in the MMU with the Program Counter value, it is possible to prevent access to the key memory by anything other than the trusted ROM routine. It is also possible to prevent reading and disassembly of the ROM code if it is made accessible only when the Program Counter executes from it.

High availability and high reliability systems are designed through use of XAP's software execution modes combined with the MMU design. For example, an attempt at illegal access to memory or i/o from user mode can trigger an error that will switch XAP into supervisor mode.

Further security can be engineered by introducing a Custom Logic Unit containing cryptography hardware, and using 'secret' CLU instructions to access it.



## Instruction set summary

The XAP5 processor has over 200 distinct instructions as follows:

- Load and Store: displacement, indexed, near, far, atomic swap (semaphores)
- Push and Pop: to and from registers, push immediate, function return pop
- Move: immediate and relative, 16- and 32-bit
- ALU operations: add, subtract, logical, multiply, divide and remainder
- CLU instructions: direct transfer to and from registers
- Compares: 8-, 16- and 32-bit with signed and unsigned result flags
- Branches: both conditional and unconditional
- Shifts and Rotates: 16- and 32-bit
- Block copy and store: near, far, immediate and register
- DSP: multiply and shift, bit flip, byte flip
- Single bit: atomic bit set and clear (for efficient port i/o)
- Miscellaneous: system calls, break, soft reset, sleep, SIF, version, flags registers, address registers, breakpoint registers

## Tools and support

xIDE is a fully-featured Integrated Development Environment for XAP. It includes a source-level debugger and XAP5 instruction set simulator. Python script support automates tasks and can extend simulation to model other parts of a chip design.

XAP5's GNU tool-chain integrates with xIDE, making the assembler, linker and C compiler easy to use.

We can provide an evaluation copy of xIDE for XAP5 that enables pre-silicon assessment of a program's code size and execution time.

Application software development can be simplified with versions of xIDE extended with software plug-ins that interface to a specific chip. These versions of xIDE can carry a licensee's brand identity and be distributed under licence.

This all makes the XAP5 attractive to professional programming teams who need to support a range of chips and applications across a variety of customers throughout a long product life cycle. Cost and investment in software continues long after a chip is designed!

## Right first time

Cambridge Consultants' patented xSIF interface provides a non-invasive debug facility capable of fully controlling the processor core and acquiring data from inside the processor and any of its memory-mapped peripherals. This allows a full investigation of chip behaviour (except where prohibited by the MMU's hardware security design).

xSIF uses a high-speed four-wire serial interface with handshake for fast debug access. Development PCs connect via an intelligent SIF pod. xSIF can also run over JTAG when used for manufacturing test.

Cambridge Consultants develops tools such as xIDE and xSIF that combine to reduce risk and time-scale for projects using XAP5.

Designers using these tools should get their ASIC right-first-time, even with programs in on-chip ROM. Our engineers use them to good effect; delivering projects on-time and in-budget. For added flexibility, XAP5 designers can include a ROM patch system whereby selected function calls are redirected to code held in RAM.

## Complete deliverables

The XAP5a is a two-stage pipeline processor core supplied in Verilog RTL for synthesis and layout. It can also be synthesised for verification in FPGA. It comes with synthesis scripts and a test bench including a complete instruction test program.

The XAP5a will clock at 220 MHz on a 65 nm logic GP process when it gives Near mode performance of 205 Dhrystone MIPS. Or at a lower clock frequency, say 16 MHz, it can deliver 15 DMIPS performance and execute direct from Flash memory.

In battery powered systems the XAP5a synthesis can be optimised for area instead of speed, or for minimum energy consumption. On 65 nm the 18k-gate XAP5a core, including all registers with routing and without scan inserted, fits into less than 0.037 mm<sup>2</sup> of silicon. Its dynamic power consumption is only 6 µW per MHz when synthesised with clock gating, i.e. a power efficiency of 155 DMIPS/mW.

Alternatively, in a lower cost 0.18G process the XAP5a core runs at up to 80 MHz in worst case conditions. The dynamic power consumption is 78 µW/MHz.

At just 18k gates XAP5a combines small size with high performance. It could replace larger 32-bit cores, saving cost and energy, or offer a migration path from old 8-bit cores.

# XAP

[www.CambridgeConsultants.com/xap](http://www.CambridgeConsultants.com/xap)

This data sheet is for XAP architecture 5.2 available for sale (in confidence) from 28 Feb 2008: the data sheet was first published 10 July 2008.

© XAP is a registered trademark and XAP1, XAP2, XAP3, XAP4, XAP5, xSIF, iSIF, SIF and xIDE are trademarks of Cambridge Consultants Ltd.

© Copyright 2008-2010 Cambridge Consultants Ltd and Cambridge Consultants Inc.

All rights reserved.

Ref: ASICs-SB-012 v2.29



**Cambridge Consultants Ltd**  
Science Park  
Milton Road  
Cambridge  
England CB4 0DW

**Cambridge Consultants Inc**  
101 Main Street  
Cambridge MA 02142  
USA