CoreCommander provides high-level functions to write data to and read data from microprocessor memory and I/O addresses without software programming. CoreCommander functions are applied via the JTAG interface.

**Applications**

CoreCommander is used in design debug, manufacturing test and (field) service for many different applications such as:

- Diagnosing "dead-kernel" boards; no embedded code is required to perform memory reads and writes.
- Determining the right settings for the peripheral controller (DDR controller, flash memory controller, I/O controller) in combination with your particular memory or peripheral device. Write settings into the controller registers and verify whether proper access to memory is possible with those settings.
- Checking the proper connectivity of a memory or I/O device by writing data to it and reading data back from it.
- Programming board (serial nr) specific data such as calibration values, a mac address or a timestamp in flash memory, or program an entire flash.

![Factsheet](image)

**Order information**

CoreComm Micro [core]  
{core} = ARM 7, ARM 9, ARM 11, Cortex-A, Cortex-R, Cortex-M, Blackfin, PXA2xx, PXA3xx, XIP4xx, PowerPC-MPC500 family, PowerPC-MPC5500 family, PowerPC-MPC5600 family, C28x, XC166, Tricore, PIC32

[1] If the uProcessor also contains a boundary-scan register then the tests and in-system programming operations can also be done using the boundary-scan register instead of the CoreCommander. Whether in that case the CoreCommander or the boundary-scan register is used depends on preference or performance.

**Background**

A uP performs read and write operations on its bus to access memory and I/O locations. The read and write cycles normally result when the uP executes a program that is stored in memory. This requires a good functioning interface between uP and its memory and a program stored in memory.

Execution of a stored program is less logical when one wants to test or debug the connections between a uP and its (external) memory. Another method as provided by CoreCommander is then needed. With CoreCommander the uP is not executing a program. Instead it is externally instructed to write data to and read data from memory and I/O addresses specified with the CoreCommander commands.
Using CoreCommander the coding to test memory connections or to program a flash memory becomes a simple and straightforward sequence of write and read commands.

**Usage**

CoreCommander can be used as interactive hardware debug tool via its high-level GUI. In this interface register access commands or full memory reads and writes can be selected for inspection and control and executed with a direct view of the results. Sequences of commands can be re-played within the interactive window or exported into a Python editor. The interactive usage is particularly valuable during hardware bring up and debugging in design and (field) service. For automated scripts the functions from CoreCommander can be called directly from programming (scripting) environments such as Python, LabView, LabWindows, Visual basic, C, C++, .NET and TestStand. This is highly valuable to create re-usable tests for specific devices or clusters and for in-system flash programming.

- Available for µProcessors, µControllers and DSP’s using one or more of the following cores:
  - Analog Devices: Blackfin
  - Arm: ARM7, ARM9, ARM11, Cortex-A, Cortex-R, Cortex-M
  - Freescale: PowerPC MPC500, MPC5500, and MPC 5600 families
  - Infineon: C166, Tricore
  - Microchip: PIC32
  - Texas Instruments: C28x (TMS320)
  - Xscale: IXP4xx, PXA2xx, PXA3xx

- Ask your JTAG supplier for the support for your specific µProcessor or µController.

Note: a CoreCommander is core specific, not device specific. Hence any µController that uses one of the above cores is supported through these CoreCommanders.