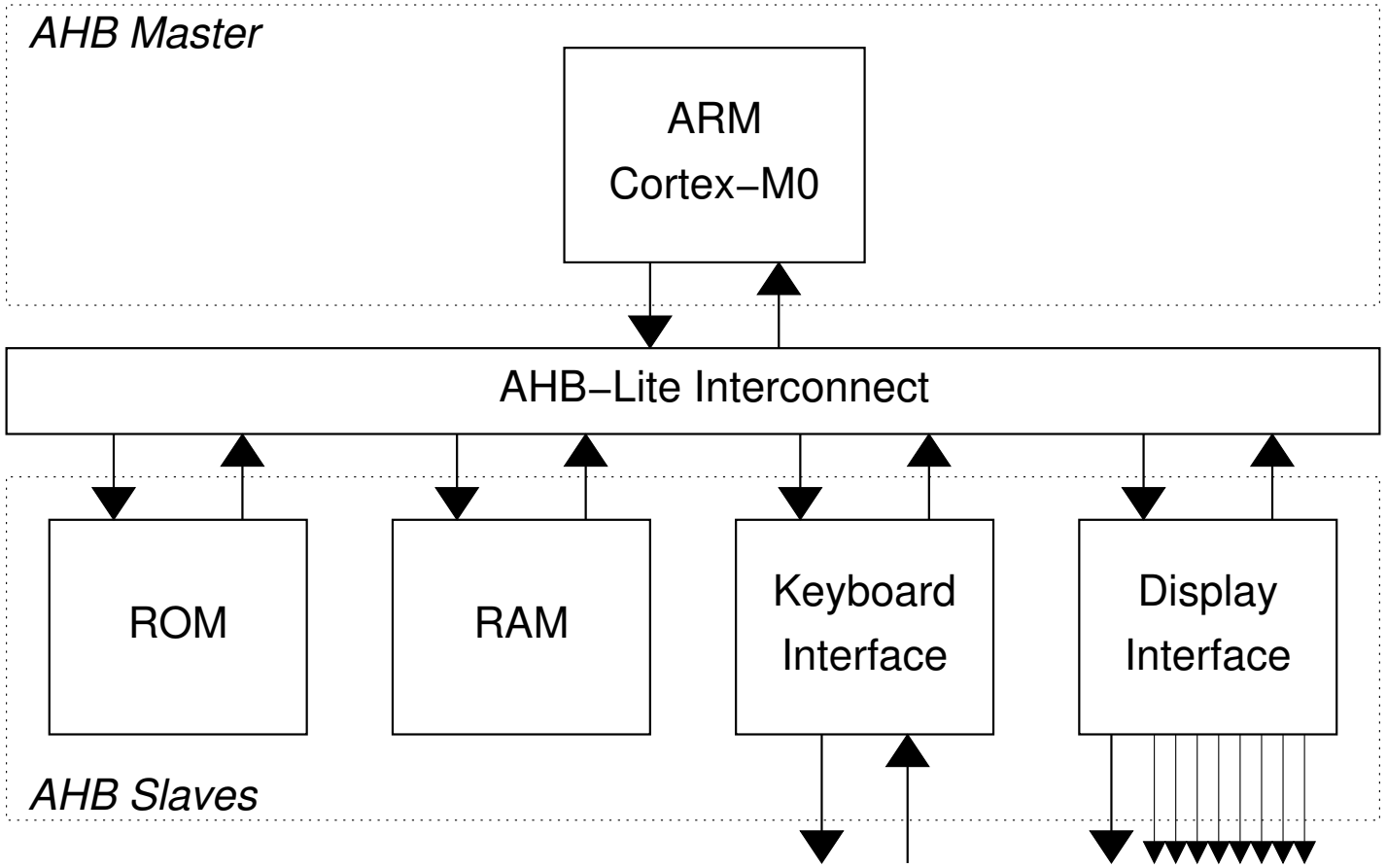
System on Chip

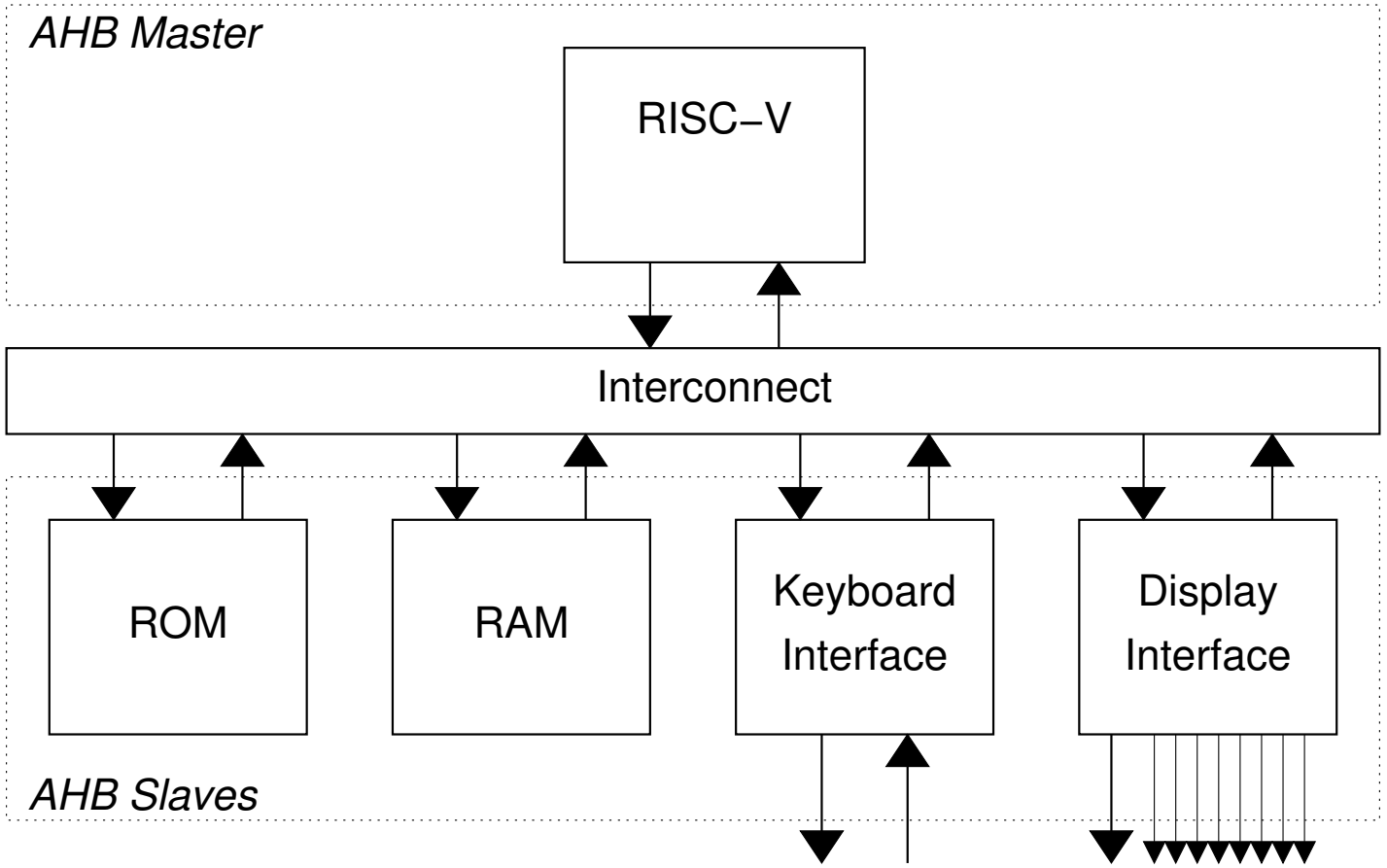
A processor based System-on-Chip will typically include:

- Hardware
  - A processor core (e.g. ARM M0 DesignStart, RISC-V PicoRV32)
  - Interconnect (e.g. AHB-Lite bus)
  - Fixed program memory (ROM)
  - Data memory (RAM)
  - Application specific interfaces (for input and ouput)
- Software
  - Application specific software (typically written in C)

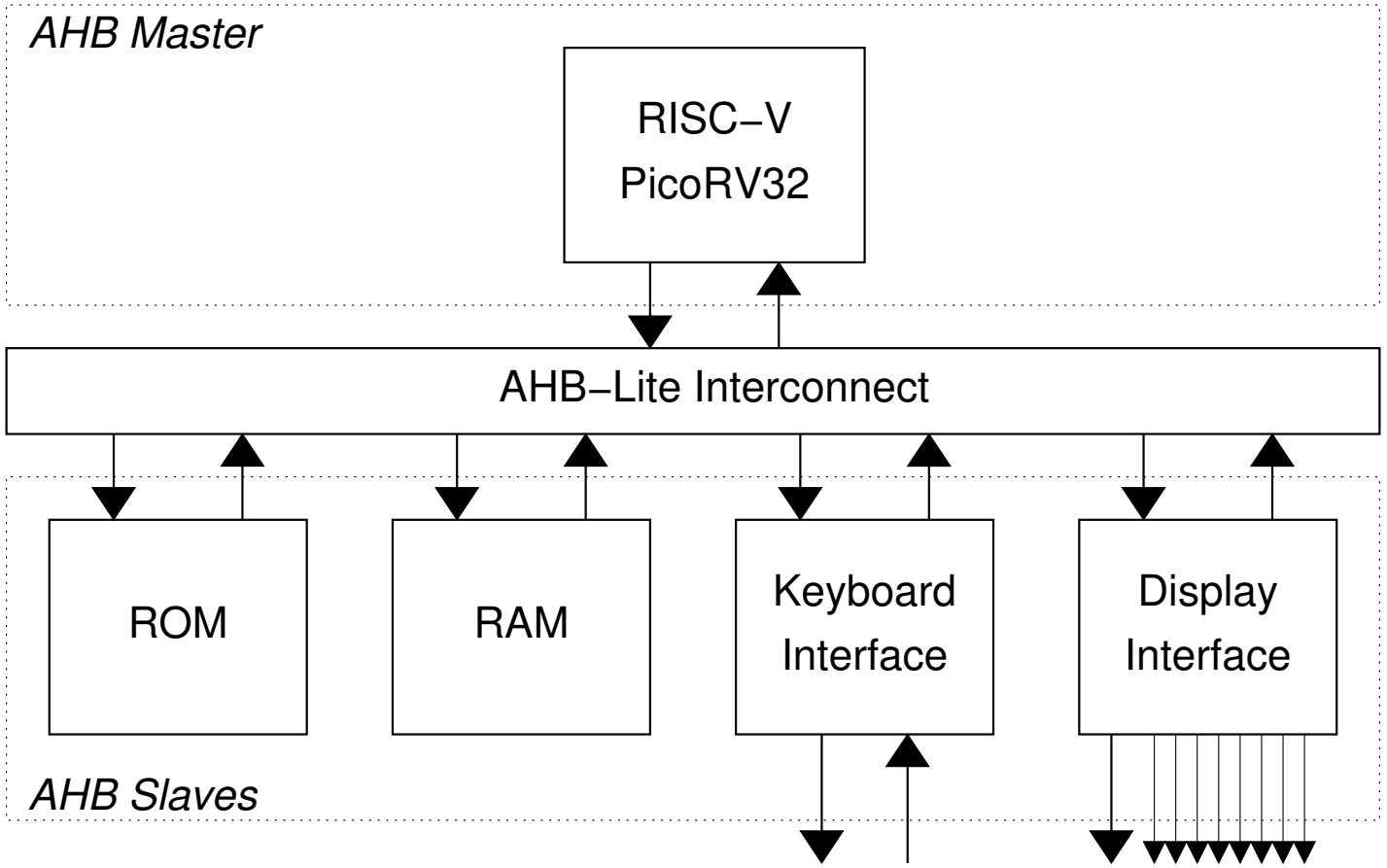
Hardware


AHB Master
ARM
Cortex–M0
AHB–Lite Interconnect
ROM
RAM
Keyboard
Interface
Display
Interface
AHB Slaves

Hardware


AHB Master
RISC−V
Interconnect
ROM
RAM
Keyboard Interface
Display Interface
AHB Slaves

Hardware


AHB Master
RISC–V
PicoRV32
AHB–Lite Interconnect
ROM
RAM
Keyboard
Interface
Display
Interface
AHB Slaves

Use of Third-Party Intellectual Property

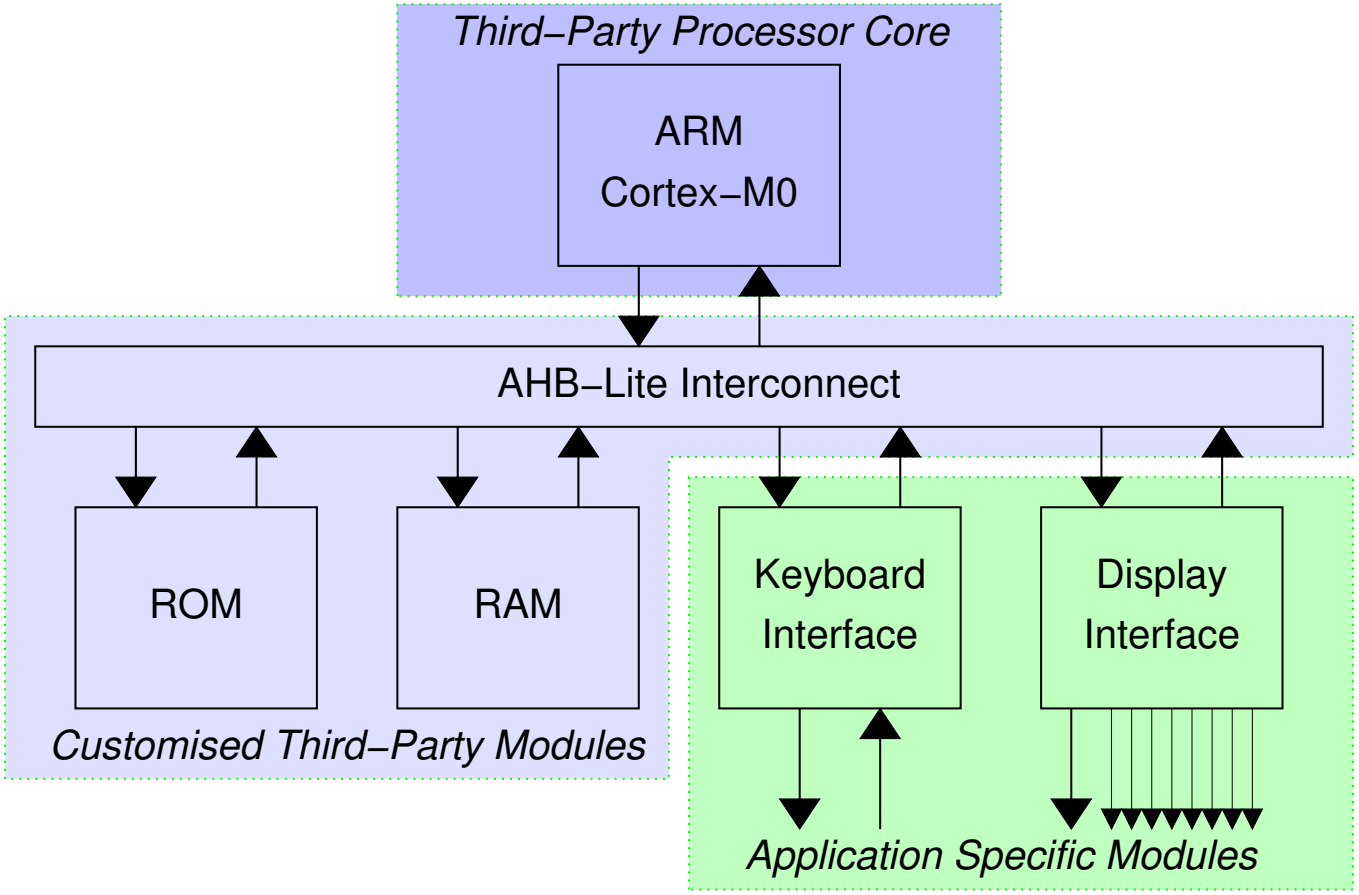
The use of hardware and software modules created by others allows designers to build larger and more complex systems.

You may make use of third-party intellectual property in your design provided that:

1. You have the permission of the designer
2. You acknowledge its use in your report
3. You do not remove copyright/licensing notices

In a simple design you might expect to use a third-party processor core together with customised versions of third-party interconnect, ROM and RAM modules

Use of Third-Party Intellectual Property



ARM Cortex-M0 DesignStart

- Simple 32-bit ARM processor core

  Designed for embedded applications.

- Executes 16-bit instructions from the ARM Thumb2 instruction set

  Supported by standard GNU compilers.

- Obfuscated Verilog Source Code

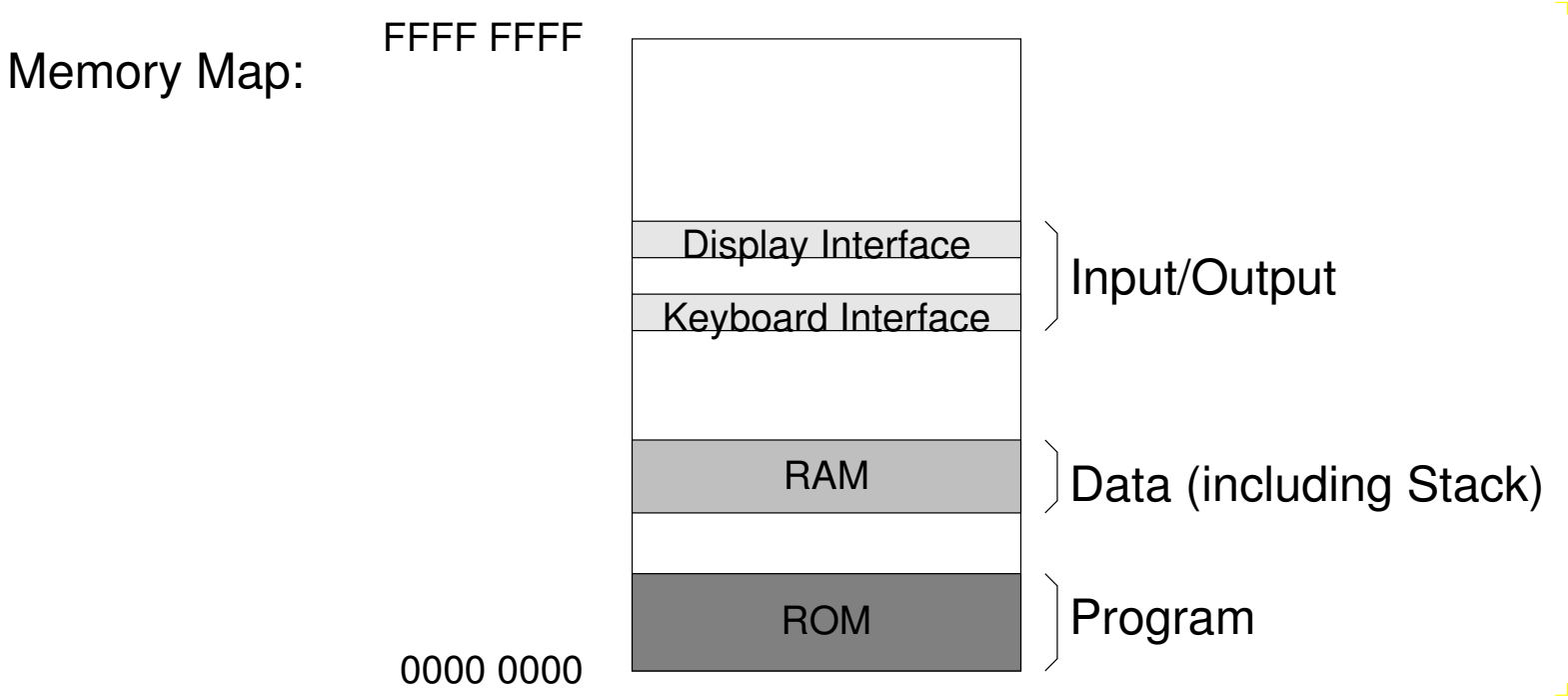
  Designed to be usable but not to be reverse engineered or customised.

- Available free for academic and other non-commercial use

  You may use the provided core for this project. If you wish the Cortex-M0 DesignStart for another purpose, you should apply to ARM for permission.

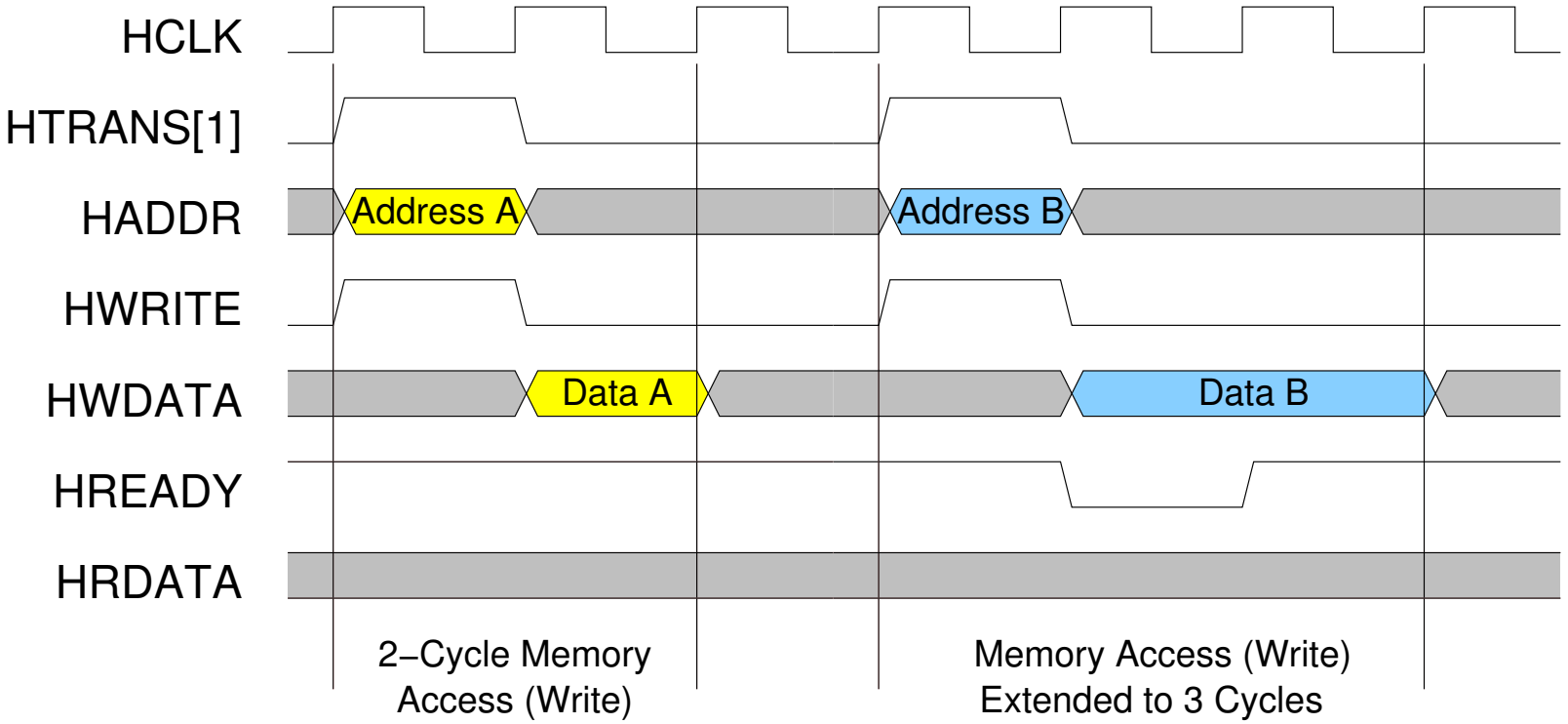
Programmer's View - Memory Mapped I/O

The programmer sees each input/output device as occupying one or more memory locations.



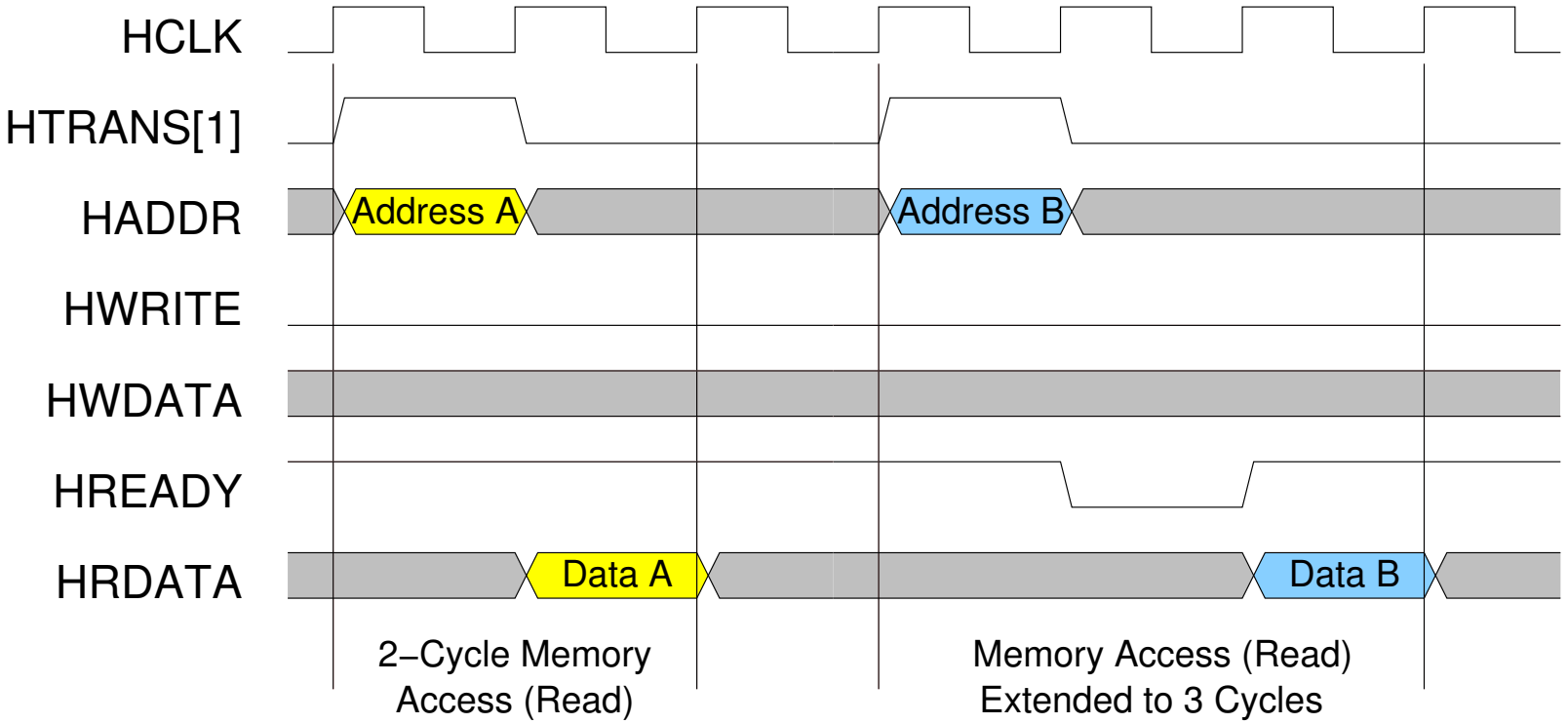
AHB-Lite Bus

Each memory access takes two or more clock cycles



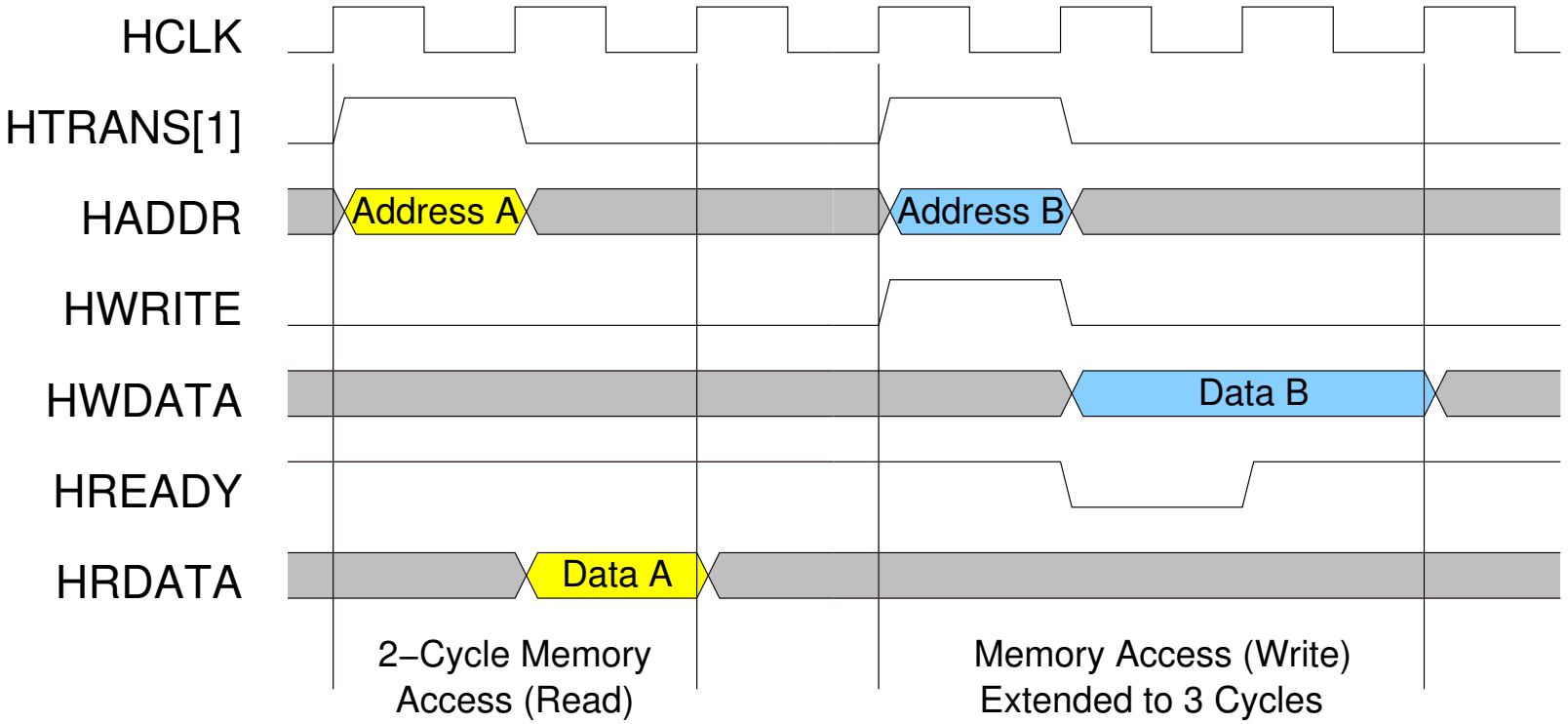
AHB-Lite Bus

Each memory access takes two or more clock cycles



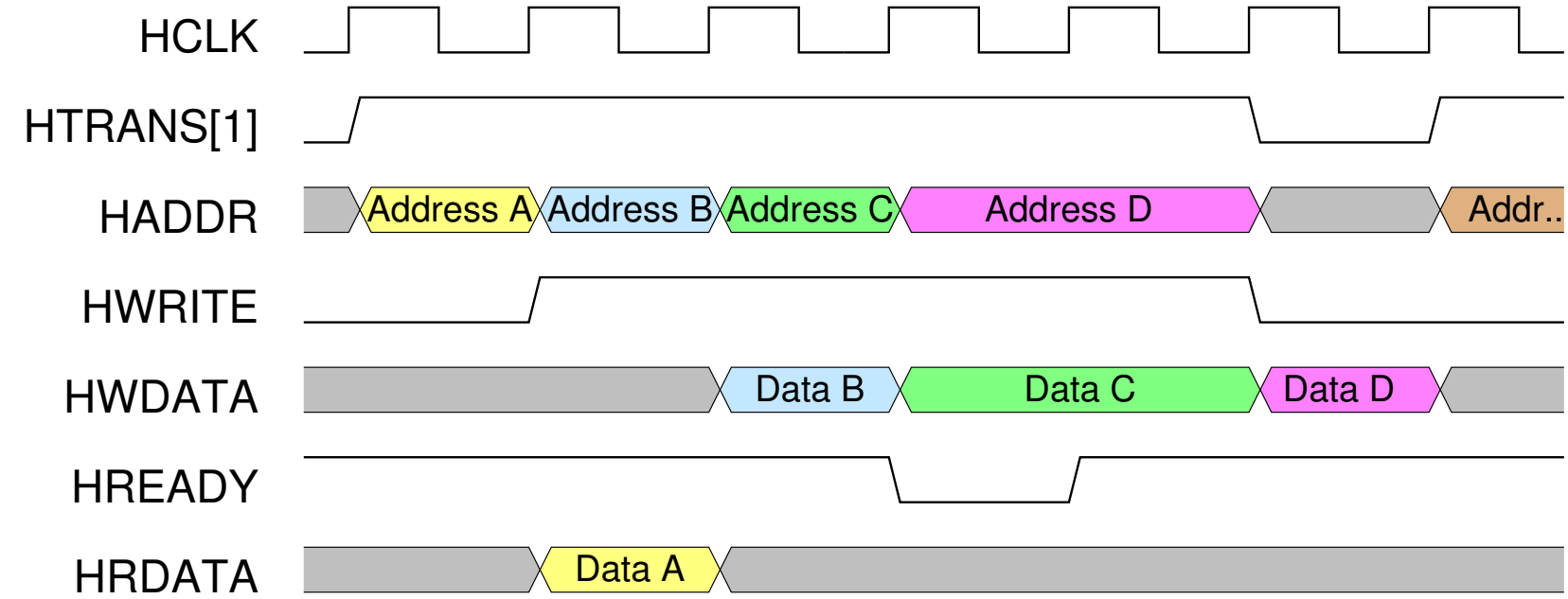
AHB-Lite Bus

Each memory access takes two or more clock cycles



AHB-Lite Bus

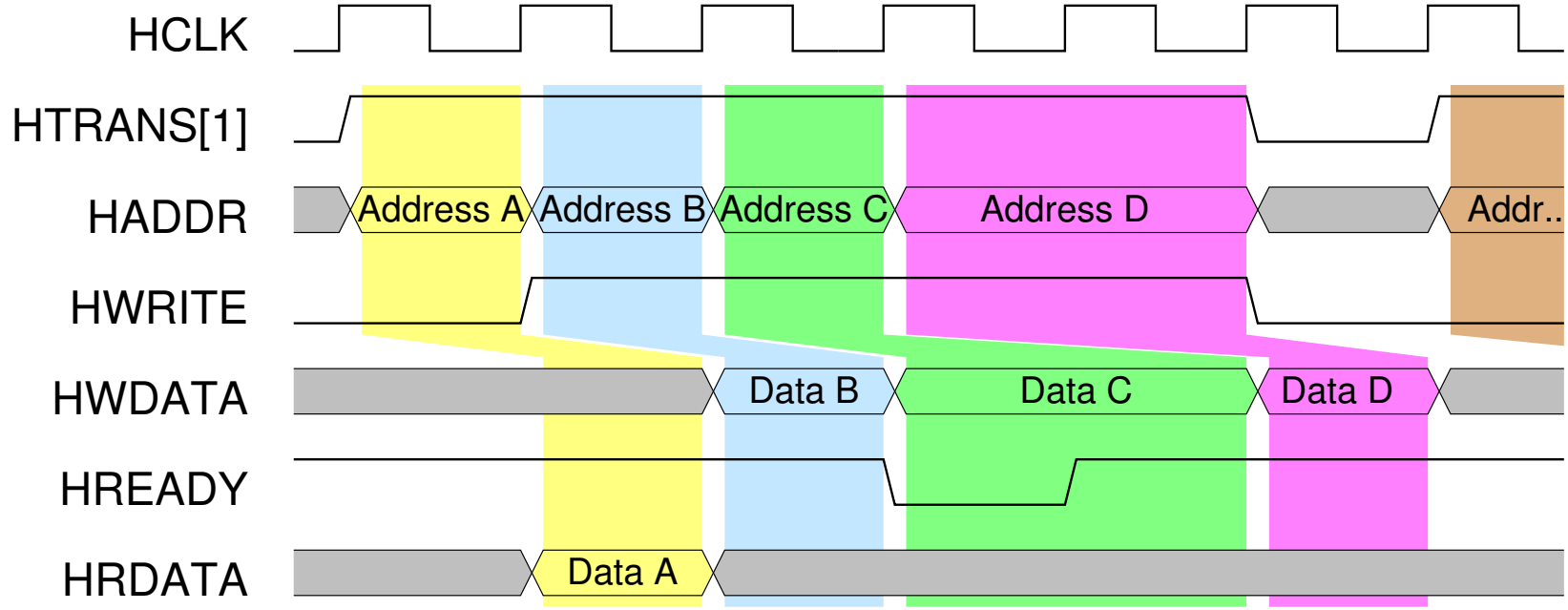
Memory accesses can be pipelined[1]



[1] the address phase of one memory access may start before the data phase of the previous access is complete

AHB-Lite Bus

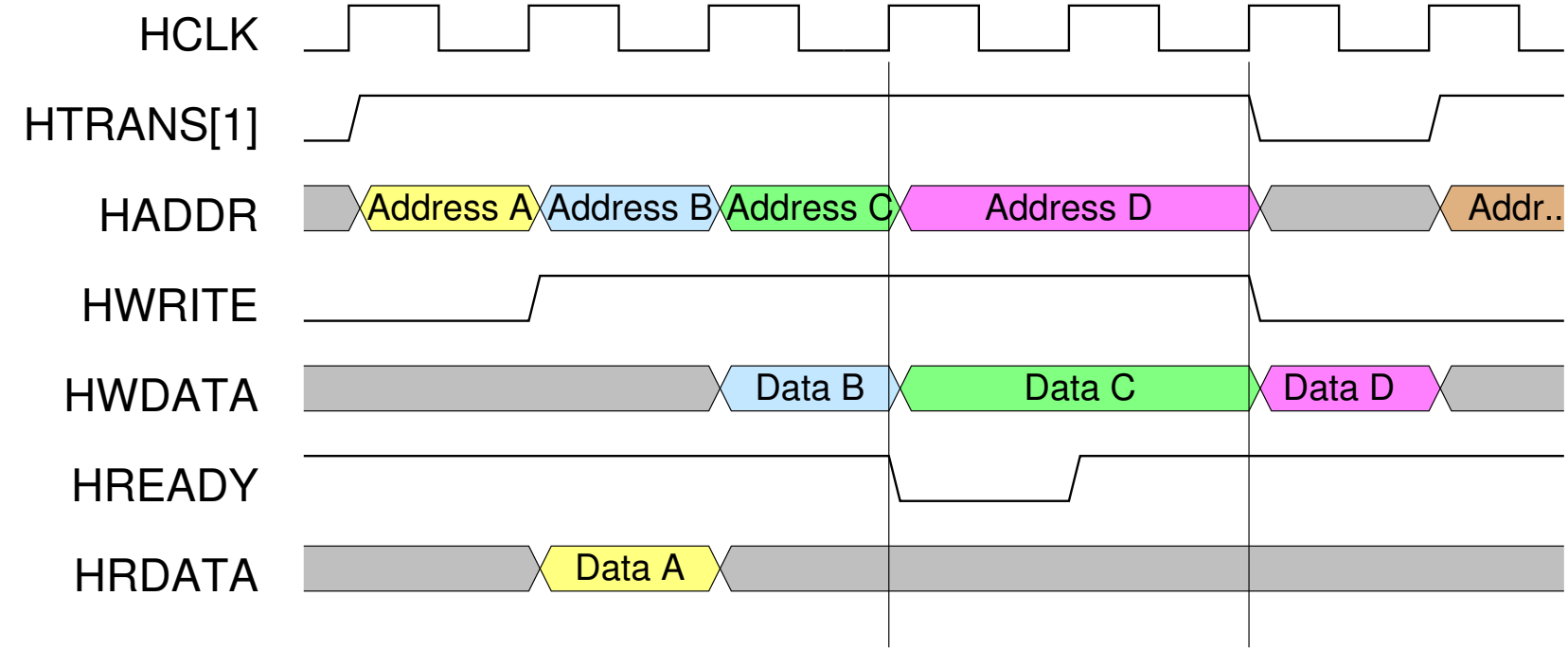
Memory accesses can be pipelined[1]



[1] the address phase of one memory access may start before the data phase of the previous access is complete

AHB-Lite Bus

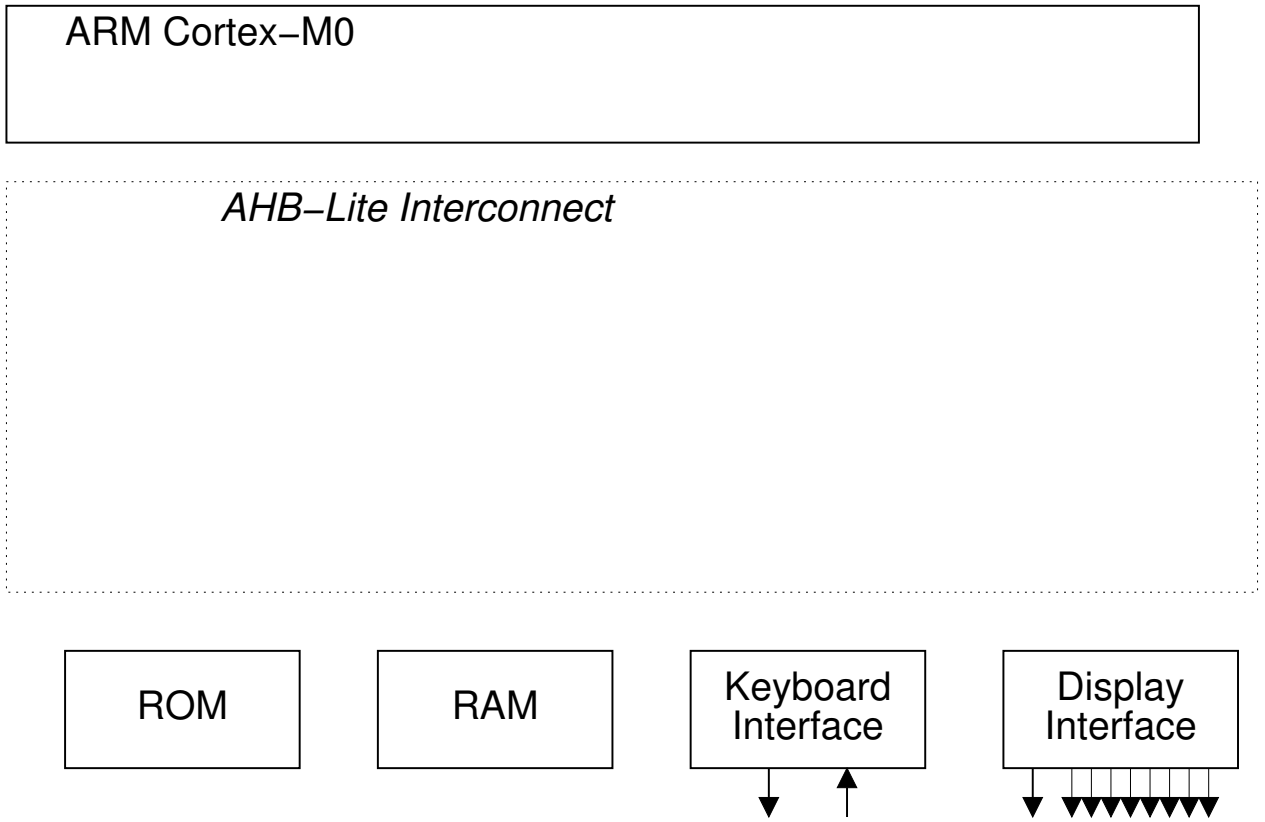
Memory accesses can be pipelined[1]



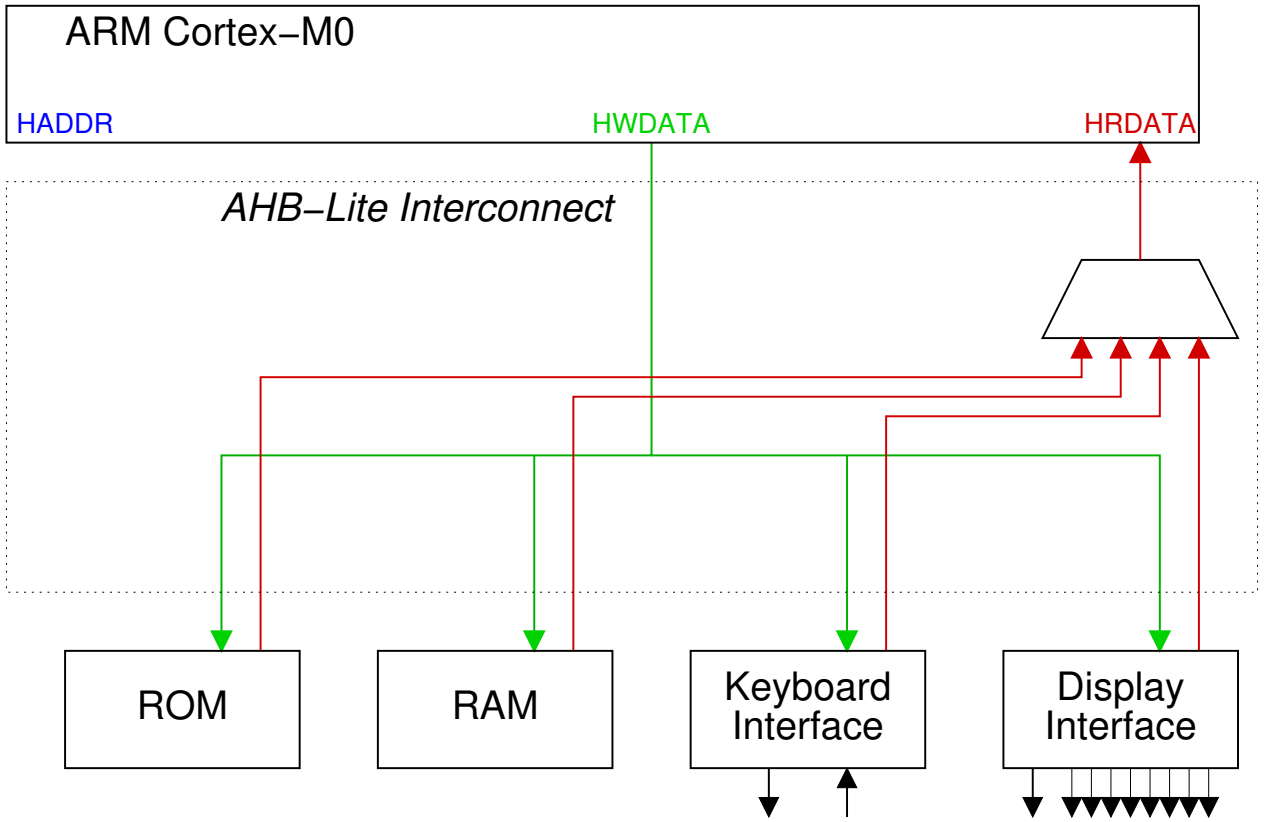
* The wait state for transaction C also extends the address phase of transaction D

[1] the address phase of one memory access may start before the data phase of the previous access is complete

System on Chip

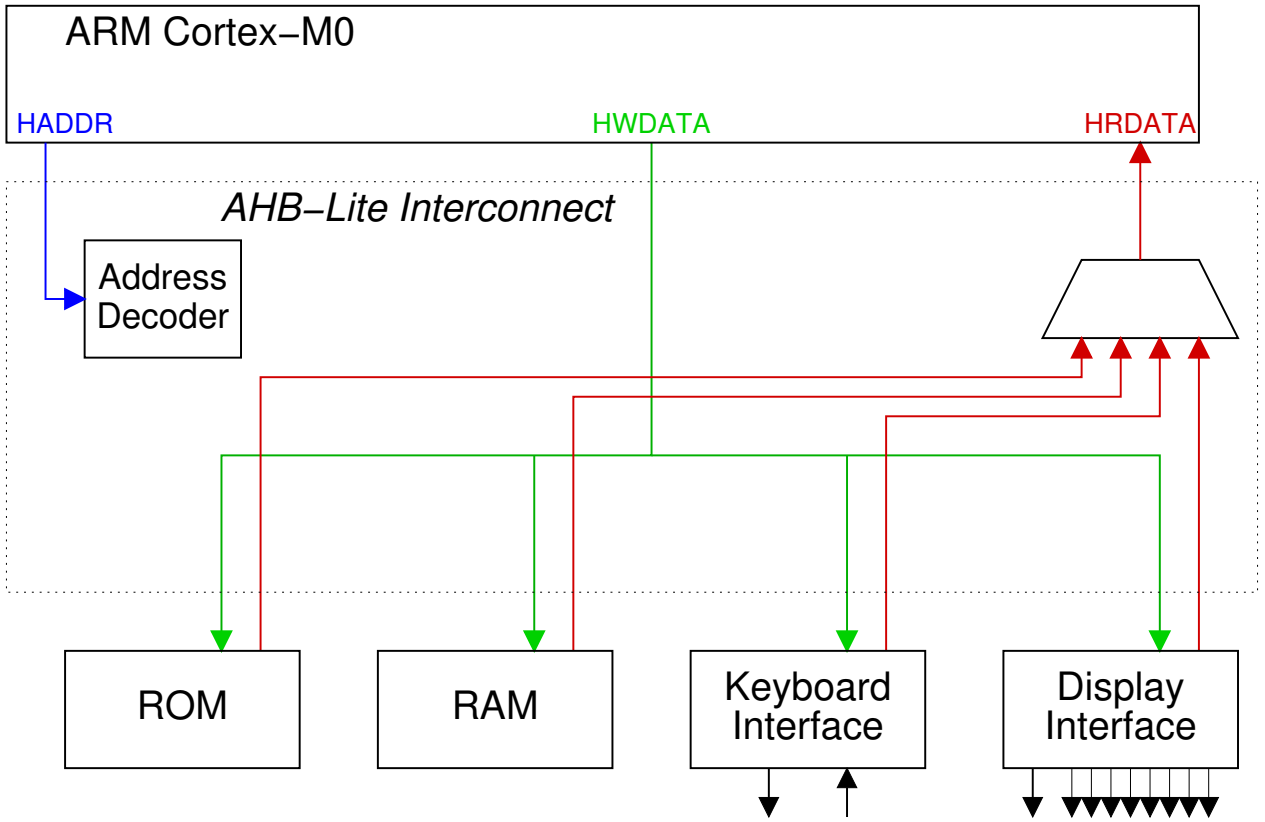


System on Chip



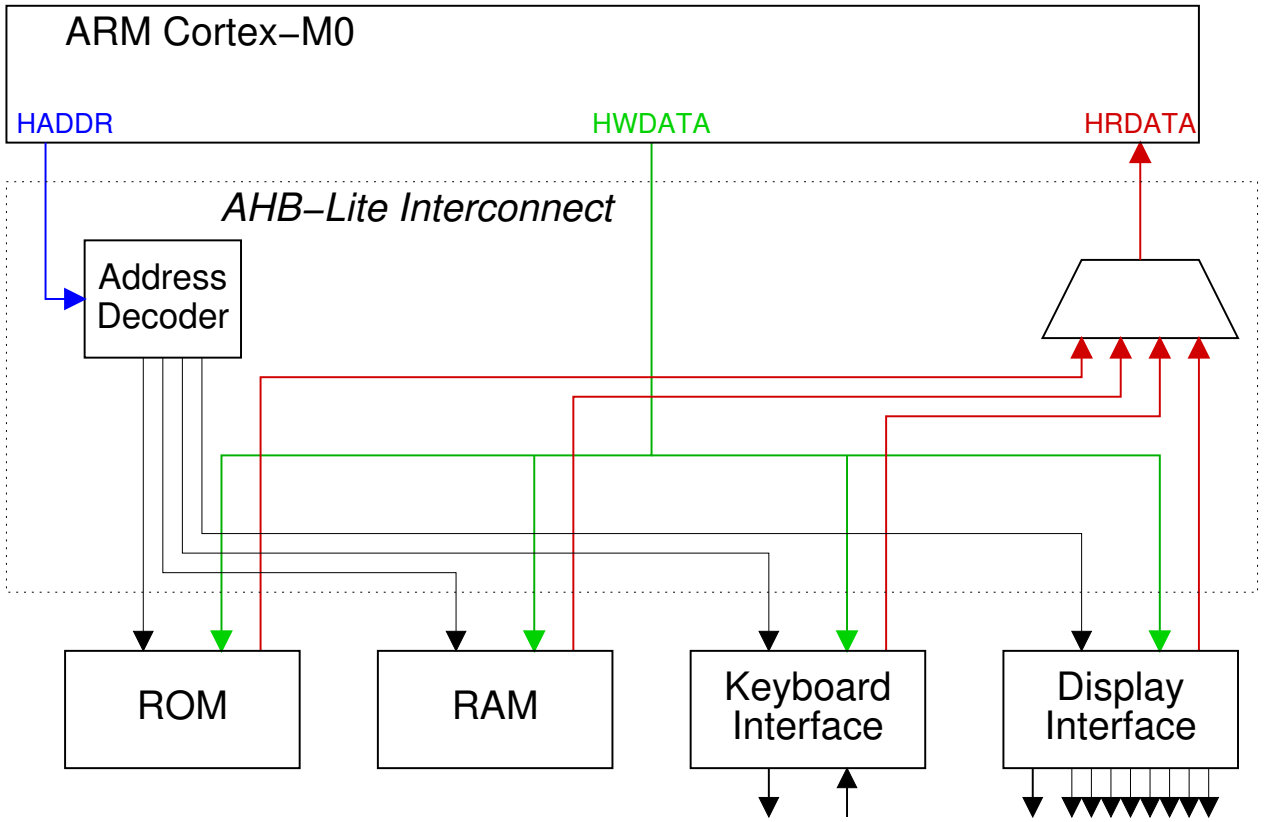
- The write data (HWDATA) is broadcast to all of the slaves
- A multiplexer selects the read data (HRDATA) from one of the slaves

System on Chip



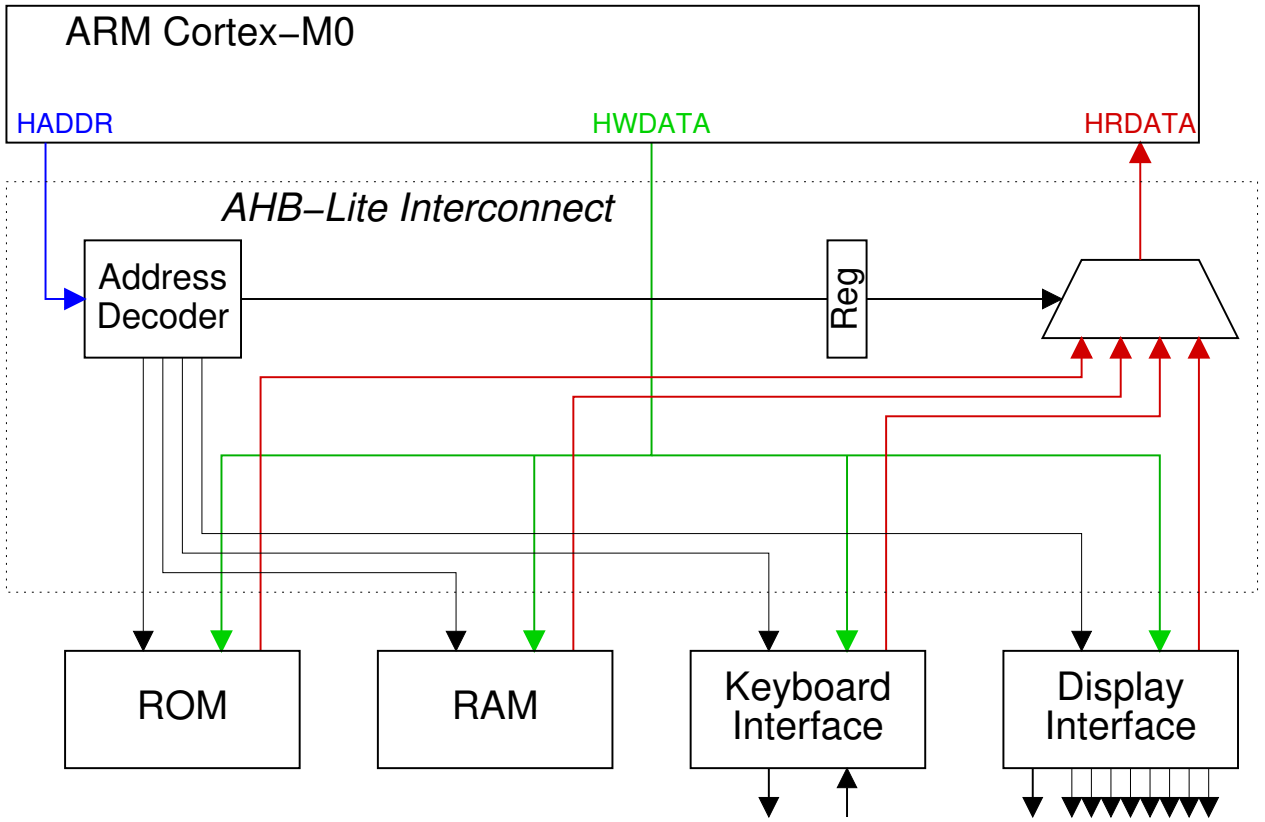
- Individual slave select signals are generated from the high order bits of HADDR

System on Chip



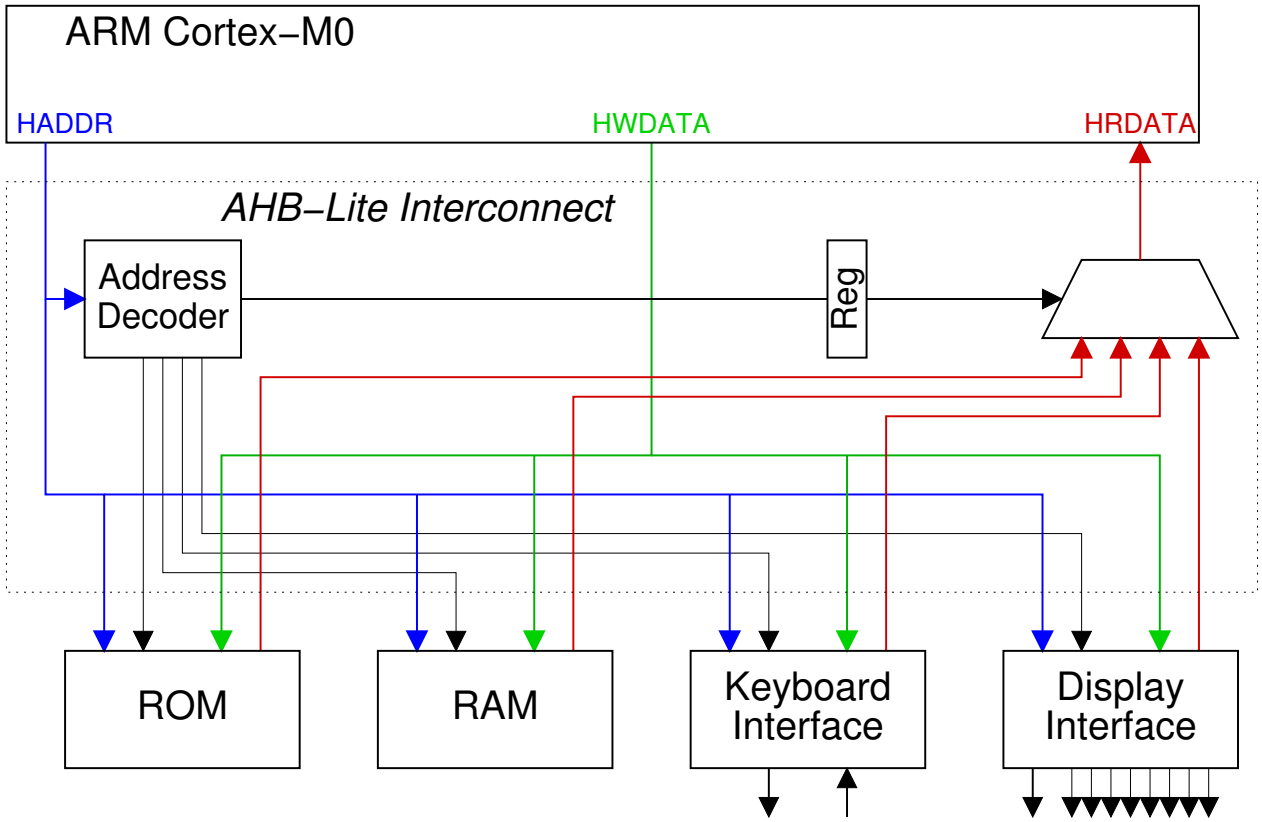
- Individual slave select signals are generated from the high order bits of HADDR

System on Chip

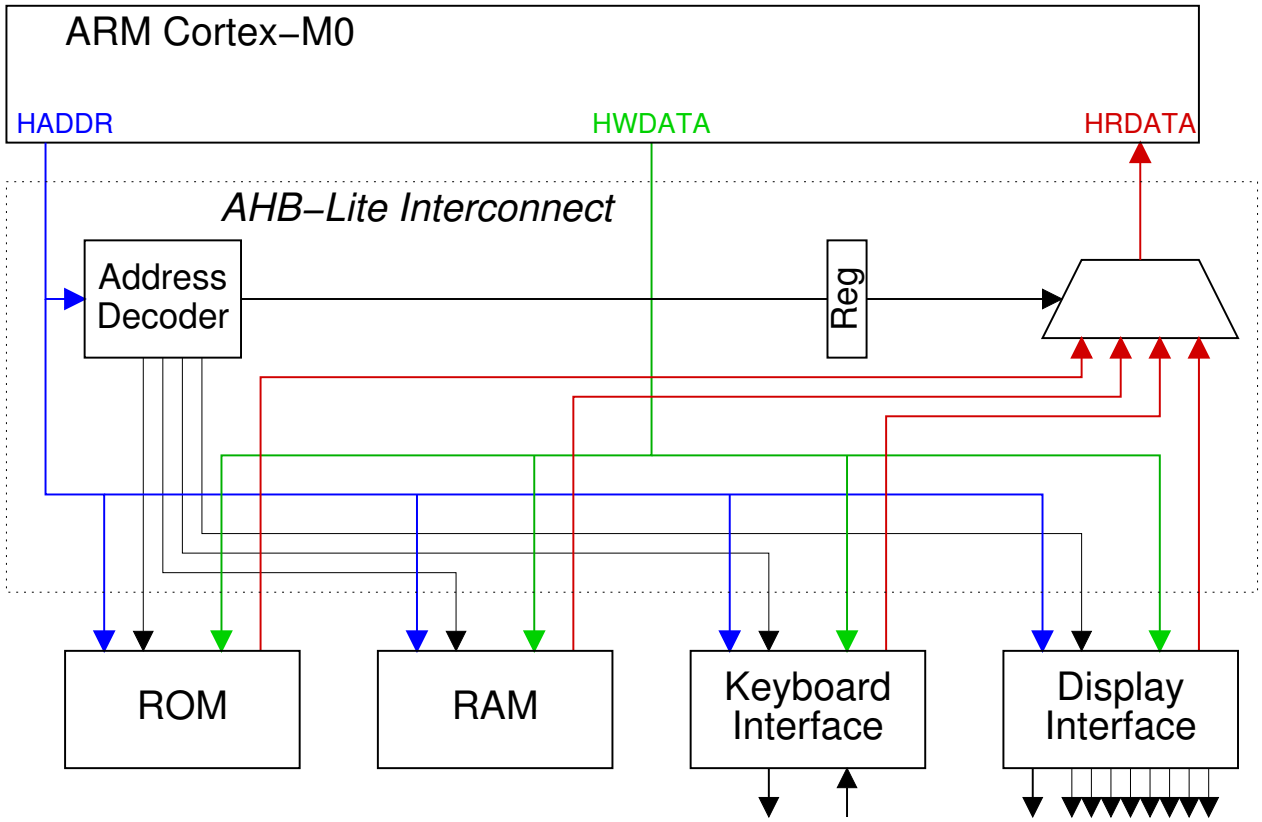


- The Address Decoder controls the multiplexer via a register which adds a single cycle delay

System on Chip



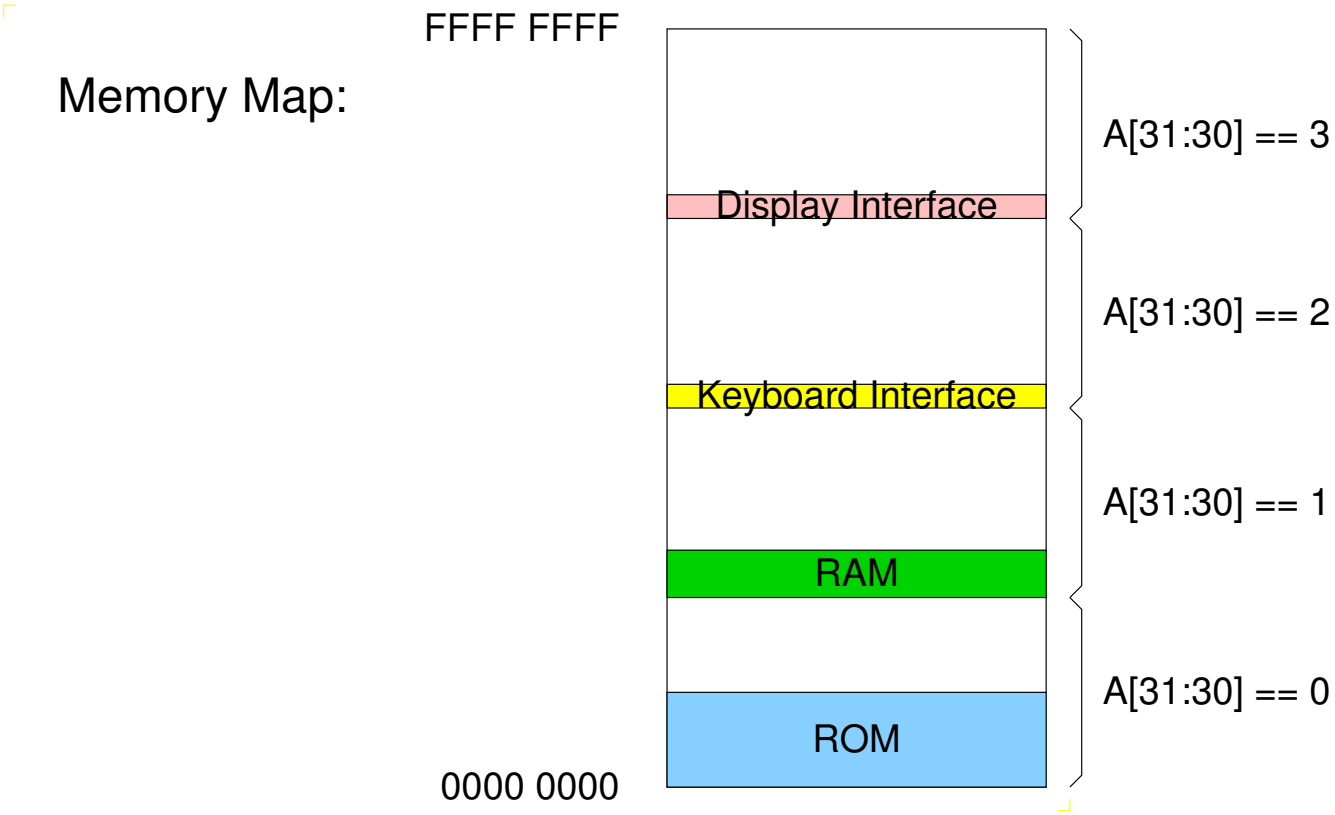
- The low order bits of HADDR are used to select registers within the slaves



AHB-Lite Interconnect

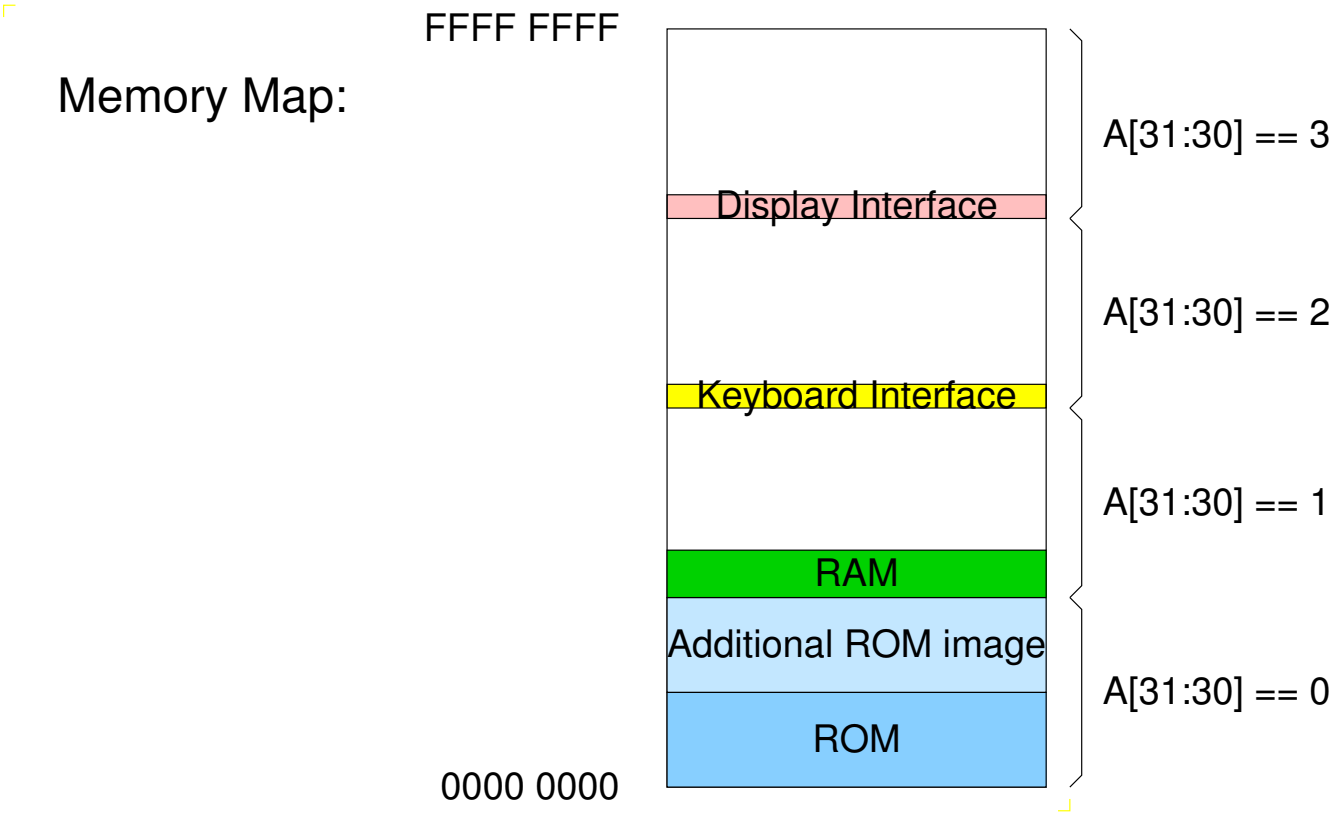
showing HADDR, HWDATA, HRDATA and individual slave HSEL select signals.

System on Chip



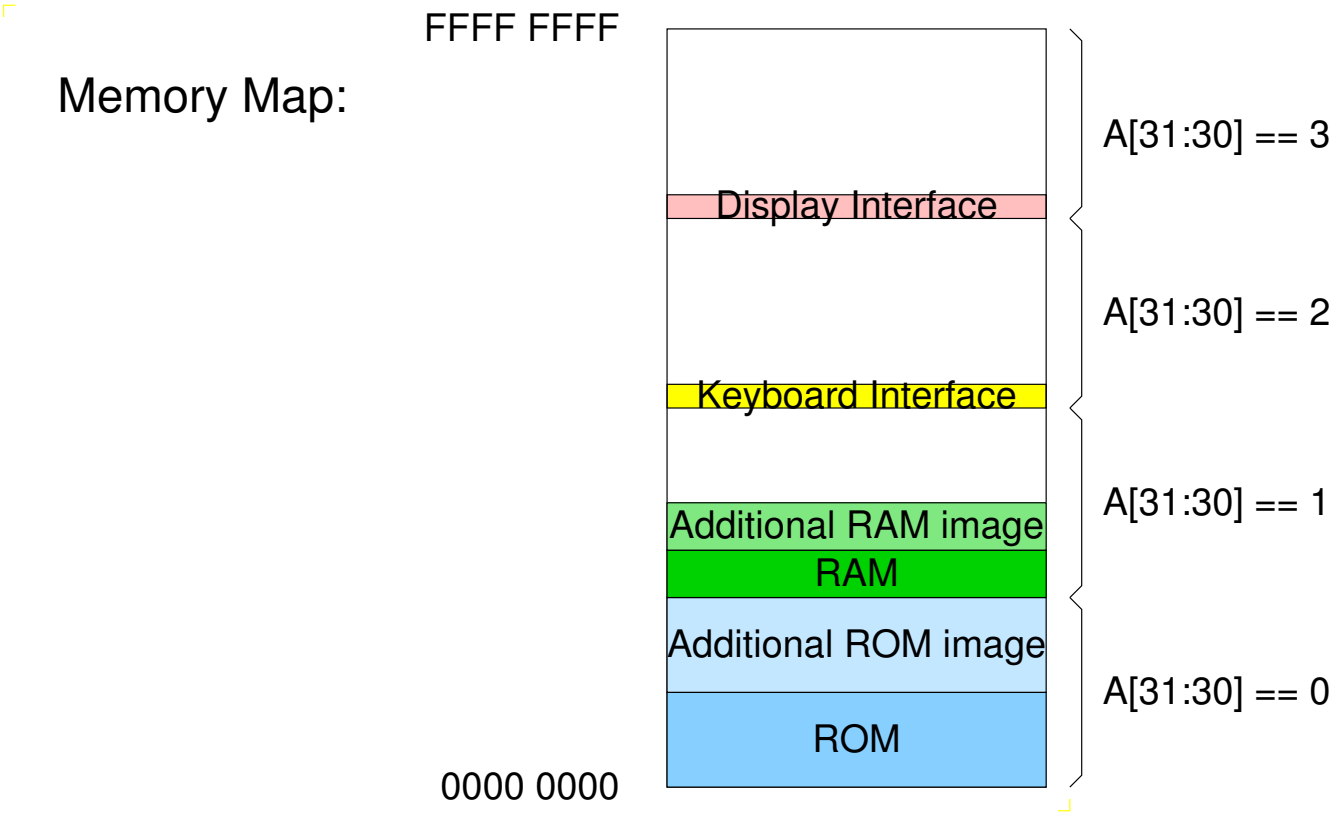
- With partial address decoding we use fewer address lines (A31 and A30 in the example above) and less logic for decoding.
- A side effect of this is that we get multiple images of some or all of the slave devices.

System on Chip



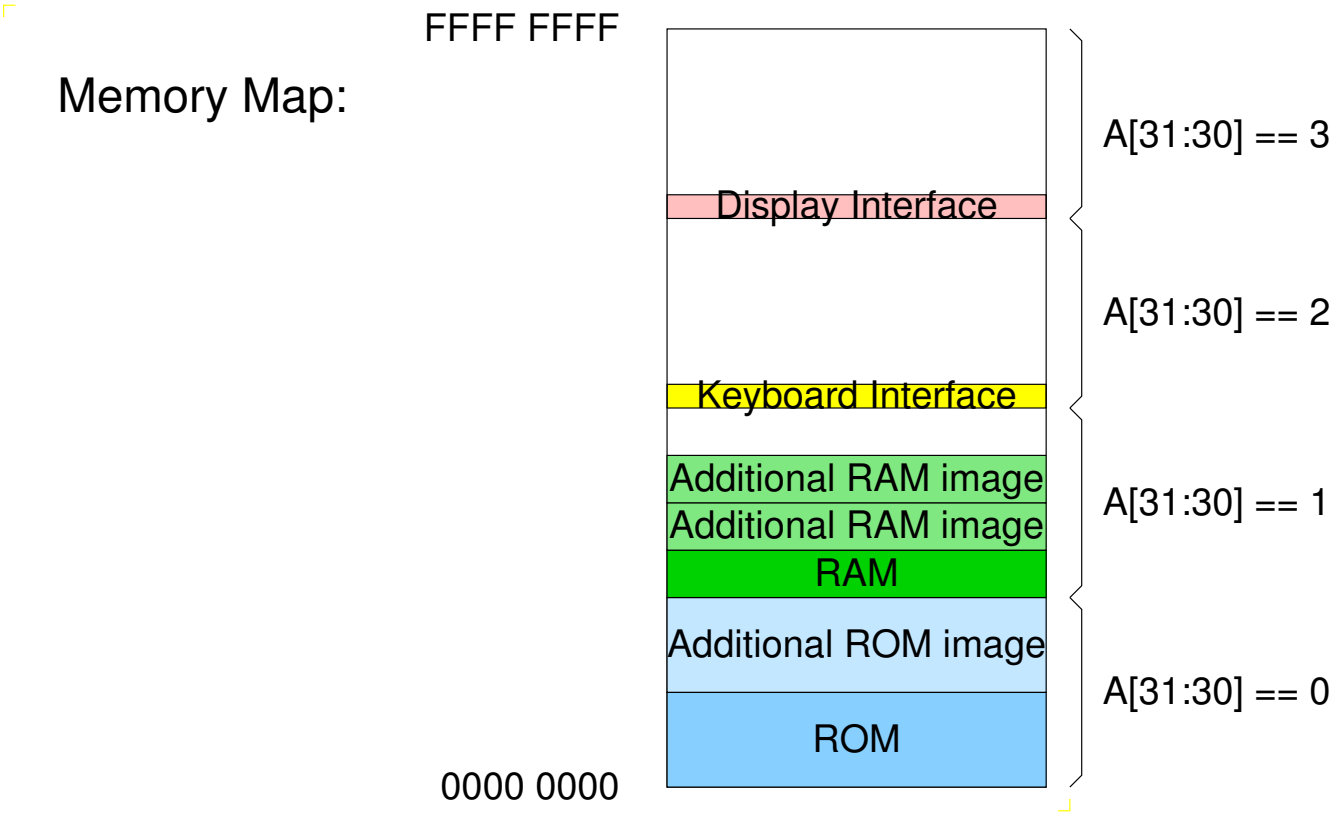
- With partial address decoding we use fewer address lines (A31 and A30 in the example above) and less logic for decoding.
- A side effect of this is that we get multiple images of some or all of the slave devices.

System on Chip



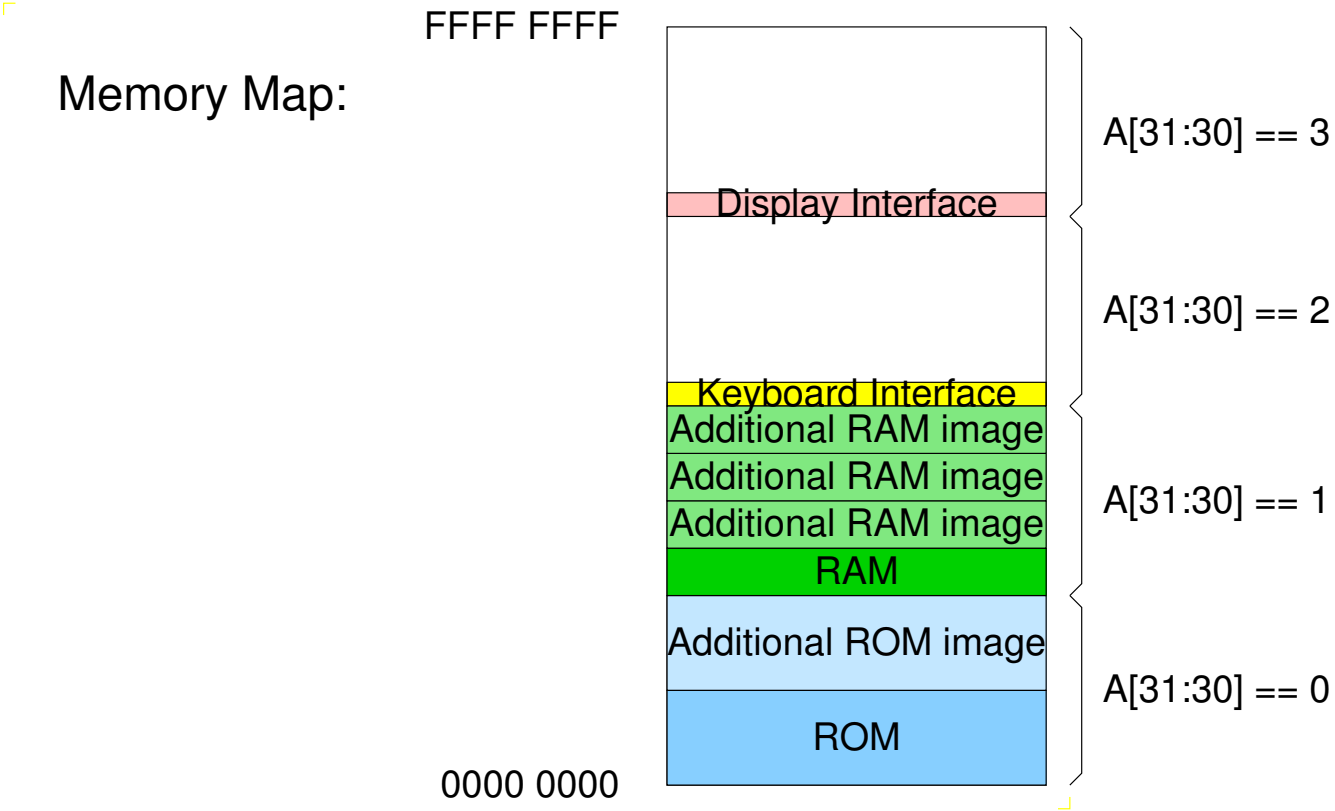
- With partial address decoding we use fewer address lines (A31 and A30 in the example above) and less logic for decoding.
- A side effect of this is that we get multiple images of some or all of the slave devices.

System on Chip



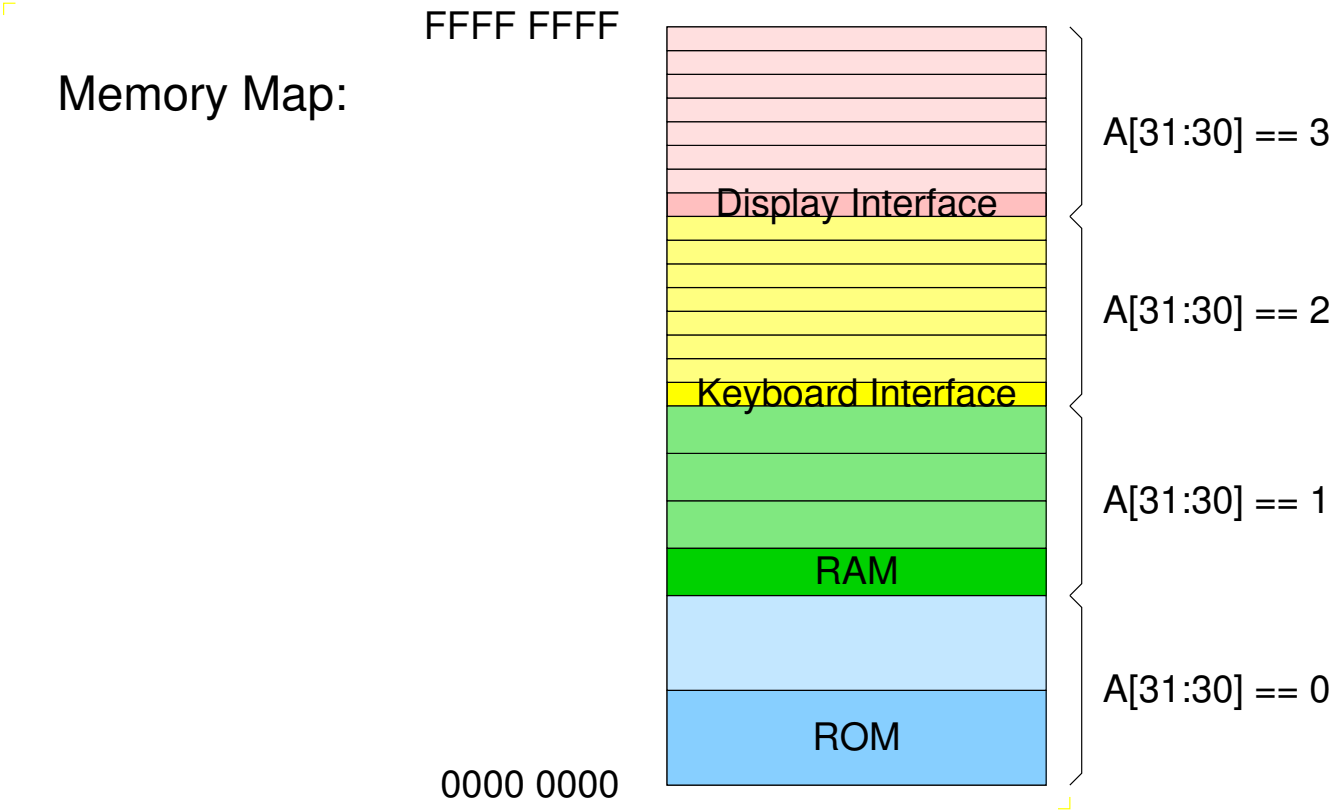
- With partial address decoding we use fewer address lines (A31 and A30 in the example above) and less logic for decoding.
- A side effect of this is that we get multiple images of some or all of the slave devices.

System on Chip



- With partial address decoding we use fewer address lines (A31 and A30 in the example above) and less logic for decoding.
- A side effect of this is that we get multiple images of some or all of the slave devices.

System on Chip



- With partial address decoding we use fewer address lines (A31 and A30 in the example above) and less logic for decoding.
- A side effect of this is that we get multiple images of some or all of the slave devices.

Application Specific Interface Module Design

- Decide on Module Function
  - Some functions are easy in hardware others are easy in software
    While non-specific input and output ports can be used (in the style of a microcontroller i/o ports), this is not usually a good use of system-on-chip resources.
- Decide on the Programmer's Model
  - What Registers?
    Input registers, Output registers, Status registers
  - What Addresses?
  - What side effects?
    e.g. Access to input or output register $\Rightarrow$ changes status

Interface Module Registers