2BA4

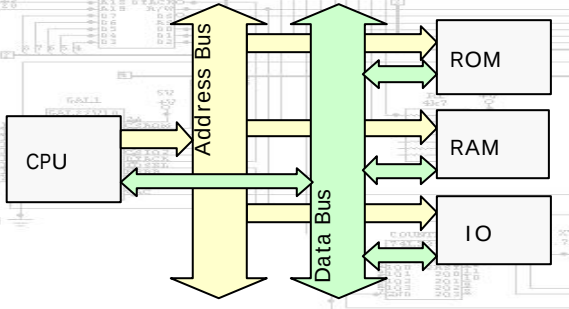
Device Selection

- Problem:
  - Identify active devices.
- Idea:
  - Use addressing in some way.
- Solution
  - Memory mapping

2BA4

Requirements

Which device provides data to the data bus during a read?



2BA4

Memory Map

- Two complementary views:
  - Map from Address to Device location
  - Map from Device to Address Range

2BA4

Map from Address to Device location

- AMap: Address -> Device x Location
- AMap($2A04C) = ($RAM_1$,$204C)
- This map is ***partial***:
  - Not every address corresponds to a device

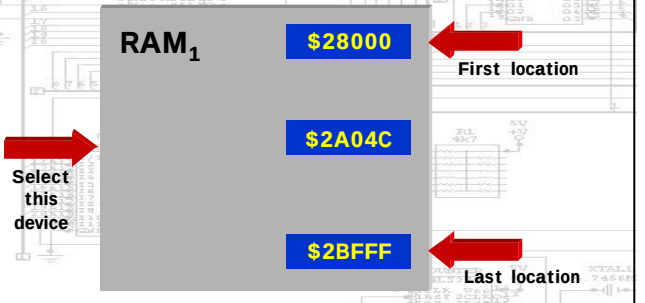
2BA4

Map from Device to Address Range

- DMap: Device -> Address x Address
- DMap(RAM_1) = ($28000,$2BFFF)
- This map is ***total***:
  - Every device location has an address.

2BA4

RAM_1



2BA4

Memory Map Representation

- We view every device as having a size
  - (No. of bytes of address space)
  - Size: Device -> N
  - Size(**RAM_1**) = $4000
- This map is total

2BA4

Base Address

- We can express a device mapping by giving a base address and using its size to determine the range.
  - Base: Device -> Address
  - Base(**RAM_1**) = $28000
- This map is total

2BA4 AMap and DMap Relationships

- We can express the following relationships.
  - **DMap(d) = (Base(d), Base(d) + Size(d) - 1)**
- If **a** is in **DMap(d)**, then:
  - **AMap(a) = (d, a – Base(d))**

2BA4 AMap and DMap Example

DMap(RAM_1)
= (Base(RAM_1), Base(RAM_1) + Size(RAM_1) - 1)
= ($28000, $28000 + $4000 - $1)
= ($28000, $2C000 - $1)
= ($28000, $2BFFF)
AMap($2A04C)
= (RAM_1, a – Base(RAM_1))
= (RAM_1, $2A04C - $28000)
= (RAM_1, $204C)

2BA4 Memory Map Requirements (Strict)

1. Unique Device per Address
   - The address range of different devices must not overlap.
2. Memory of the same type & purpose has contiguous range.
   - Type = {ROM, RAM}
   - Purpose = {User, System, Display,…}
3. Special Device-Specific Requirements met.
   - (e.g. Processor Power-up/Reset vector)
     - ROM should be mapped to cover addresses where start-up vectors are expected by CPU.

2BA4 Memory Map Requirements (Not so Strict)

4. Unique Address per Device Location
   - Allow small non-Memory devices to ignore this rule.
     - Typically I/O devices
- Relaxing Requirement 4 makes **DMap** multi-valued.

DMap($IODev_1$)
= {($E000, $E007),…,($EFF8, $EFFF)}
Where Size($IODev_1$) = 8

2BA4

Memory Map (Design)

- Must meet "System" requirements.
  - ROM, RAM size, No. and type of I/O devices, …
- Must satisfy Memory-Map "requirements".
- Should allow efficient implementation.
- Should cater for system expansion.

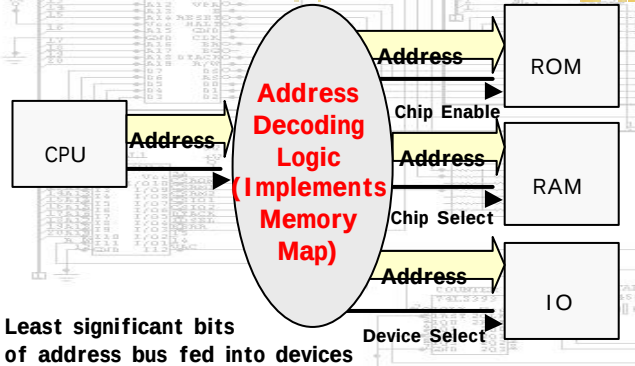
2BA4

Memory Map (Implementation)

- Problem, Given Memory Map:
  - Need to use Address to select a device and location within it.
- Solution:
  - Address Decoding Logic
  - Device Chip Select Inputs
  - Some Address Bits fed to the Device
  - Address Strobe has vital role.

2BA4

Address Decoding



2BA4

Address Decoding Logic

- What does it do?
- Input:
  - The Address Bus and Strobe
- Output:
  - A **C**hip **S**elect line (**CS**) for every device.

2BA4 Address Decoding Behaviour

- Behaviour:
  - At most one CS line is asserted.
  - If VALID ADDRESS maps to corresponding device.
- Valid Address:
  - Contents of Address inputs, when strobe is active.

2BA4 Tri-state Pins

- Only one tri-state pin should drive a bus line at a time.
- CS lines enforce this behaviour.
  - Only one line is asserted at any time.
- MEM chips keep data pins in Hi-Z state when their CS is not active.

2BA4 Address Decoding Logic (How)

- How does address decoding do it?
- Determining selected Device
  - -> Look for higher-order address bit patterns.

2BA4 Example:

- RAM_2 occupies $20000…$2FFFF
  - Chip Select for RAM_2(CS_{R2}) active when $\overline{A19}…\overline{A16}$ = $2

CS_{R2} = A19 and A18 and A17 and A16 and $\overline{AS}$