VME64x in CMS

a proposal for some design rules

http://cmsdoc.cern.ch/~cschwick/VME/index.html

please comment

- Why design rules
- Interesting VME64x features
- Design rule requirements
- 7 rules
- what does this cost

Why design rules

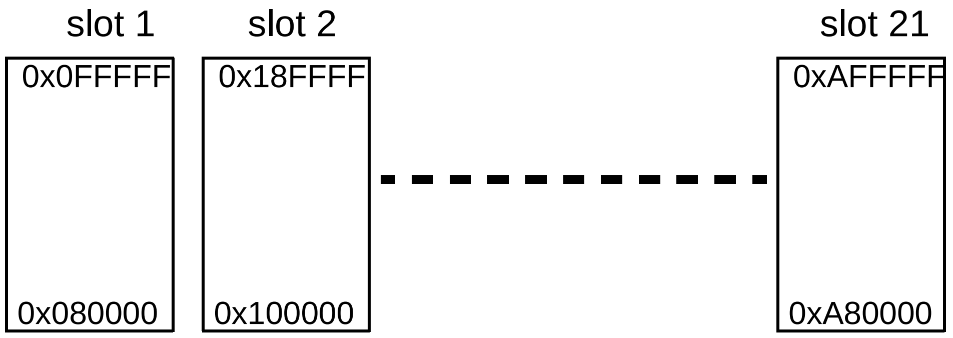
- CMD group can provide general sofware package with high functionality
  - automatic module recognition
  - automatic module identification (not only the type but the serial number)
  - automatic address space configuration
  - automatic capability discovery (which access type is allowed?)
- Hardware maintenance becomes easier
  - no address jumpers to be set
  - module replacement becomes “plug and play”
  - important for system reliability
  - important issue when experiment becomes old and “experts” are not around anymore
- Software maintenance becomes easier
  - more common packages which need less system specific code

Interesting VME64(x) features

- New accesses types
  - Data width up to 64 bit
  - Multiplexed mode (uses Address bits for data transport -> 64 bit on J1/J2)
  - Retry capability
  - Lock capability
  - NO BROADCAST, though (but there exists an extension)
- Configuration ROM / Configuration Status Register (CR/CSR)
  - similar to PCI
  - provides Plug and Play capability (see below)
  - provides capability discovery
  - provides unambigous module indentification
  - provides user defined CR or CSR regions

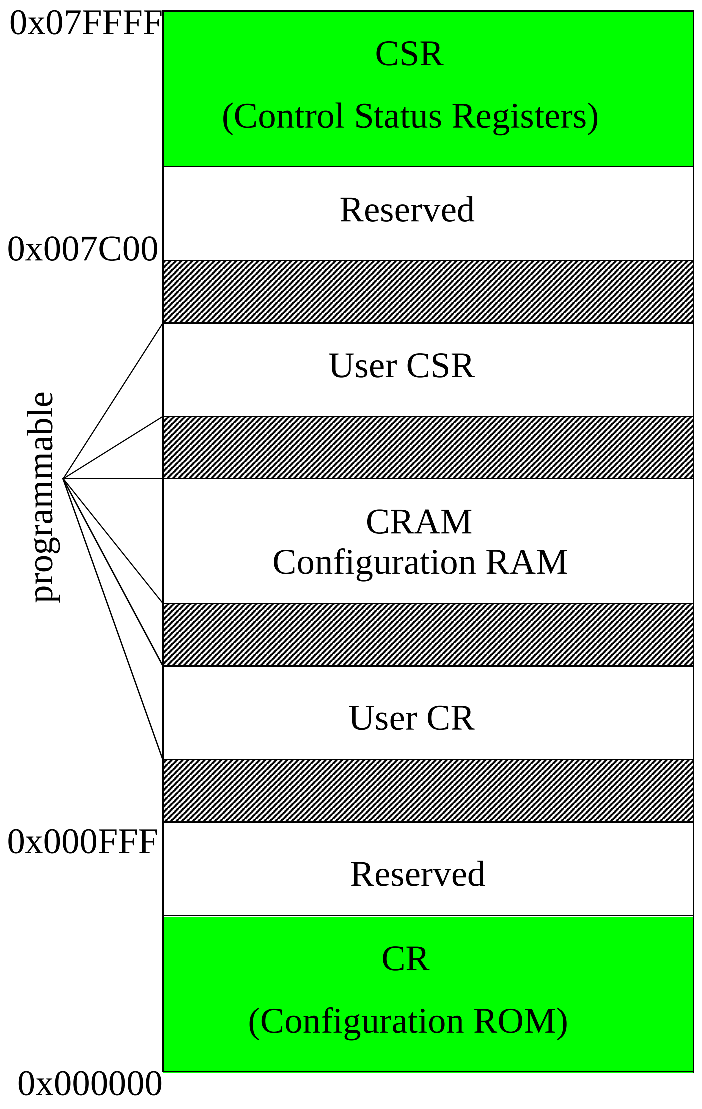
The Configuration ROM / Configuration Status Register : CR/CSR

- Access to the CR/CSR via dedicated A24 / AM 0x2F / D08 (O or EO) D16 or D32.
  - Every modules occupies a 512 KB CR/CSR space



0xF00000 is “amnesia address”,
0x000000 is forbidden

  - mapping of the CR/CSR into A24 address space defined by geographical address (slot number)
  - geographical address available in each slot via 6 newly defiend pins
  - CSR is RAM region
  - CR is ROM region
  - user defined regions are defined by pointers in CR.



Plug and Play in VME64x

- VME64x Plug and Play is very similar to PCI
- VME64x defines up to 8 logical functions per board
  - (the precise meaning of a function is not further defined...answer from VITA is pending)
- Possible Plug and Play procedure

1) Scan the VME bus for VME64x modules (by try to access configuration space identifier)
2) For each module found:
   - for each function defined
     - read out access capabilities (possible AM, data width)
     - read out required address space
3) Build address map
   This puzzle must take into account VME and VME64 modules in crate (the baseaddresses of these are “hardcoded”)
4) Set the baseaddress of the module
5) Enable the module
6) Identify modules and configure them apropriately

Requirements to Rules

- Allow for Plug and Play
- Allow for unambigious module identification
- Leave maximal freedom to system designer
- Do not complicate the design of the module
- Allow for mixed systems: “old” VME - VME64x

7 golden(?) rules

1) Implement the CR / CSR space of VME64x specification
2) Implement the serial number
   - possibly define a format for CMS modules
3) Implement the plug and play capabilities with “address relocation” as defined in the VME64x spec.
4) Do not use dynamic function sizing
   - CMS modules “know” how much address space they need
5) Do not use fixed base addresses
6) Use reprogrammable media (e.g. Flash) for CR
   - eases debugging, new feature implementation,
7) All other VME64x features are optional and MAY be implemented

Cost of these rules

- Need to implement CR/CSR space access
  - This is not a special access => easy implementation
- Implement Address Relocation for Plug and Play
  - address comparators need to use registers and not hardcoded input
  - same for Address Modifier comparation
- All other recommended features are implemented by CR programming (ROM programming)