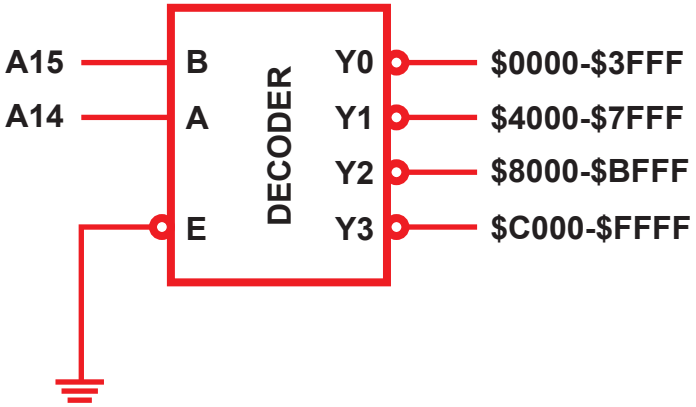
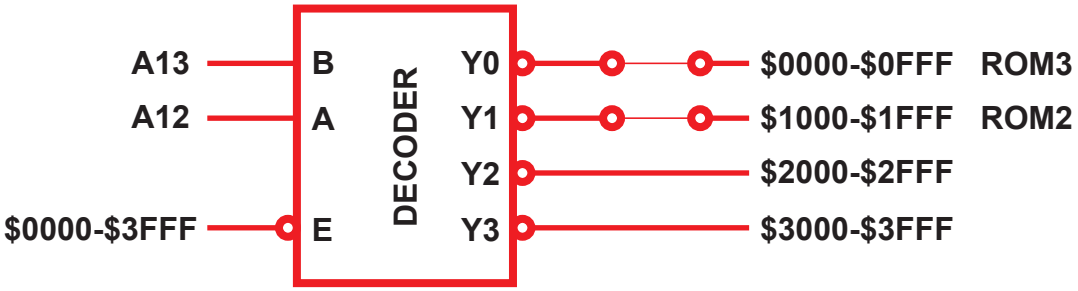
Analysis of the
Aamber Pegasus Memory Map Decoder

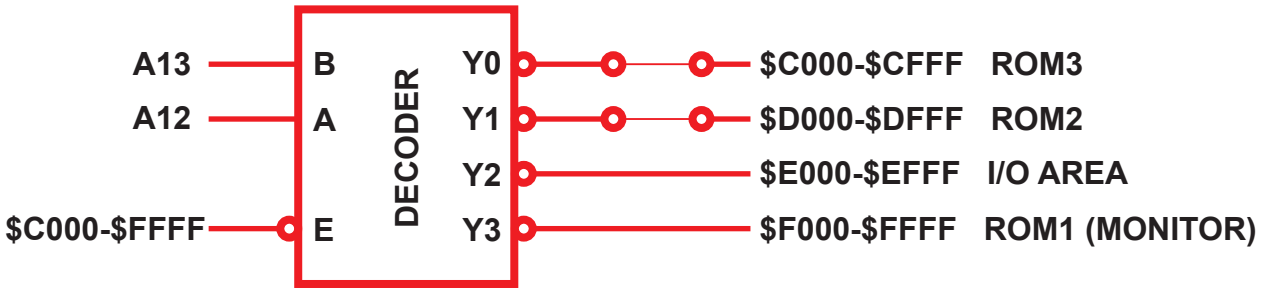
by: J&R of http://www.GIMEchip.com/



Initially, the memory map is divided into four 16K sections.



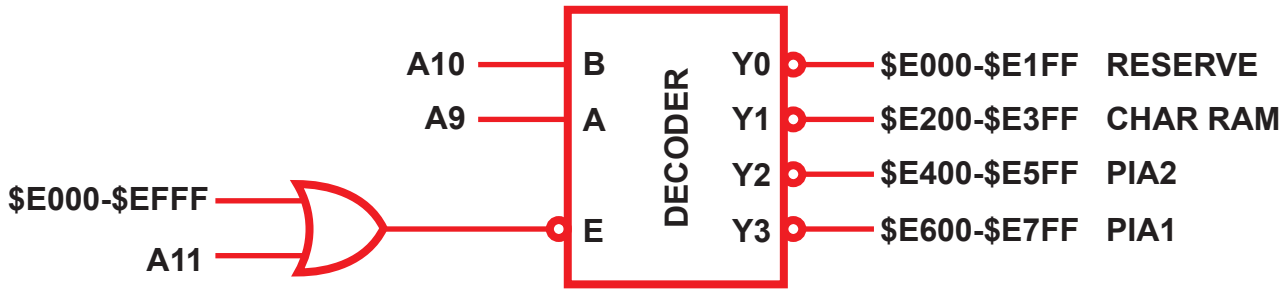
The lowest 16k is divided into four 4k sections. The Aamber Pegasus schematic only displays the $0000-$0FFF and $1000-$1FFF decodes, however, the $2000-$2FFF and $3000-$3FFF are decoded and may be tapped from the proper pins of the decoder chip. Note the use of the two jumpers - this allows the ROM2 and ROM3 decodes of this section to be disabled. This is because the design of the pegasus allows them to be mapped alternately at $C000-$CFFF (ROM3) and $D000-$DFFF (ROM2).



Here, the upper 16k is also divided into four 4k sections. The Aamber Pegasus schematic only displays the $C000-$CFFF (ROM3), $D000-$DFFF (ROM2) and $F000-$FFFF (ROM1) decodes. The I/O Area is, however, decoded and may be tapped from the proper pin of the decoder chip. This area ($E000-$EFFF) is, in fact, used in conjunction with another decoder to further decode the I/O AREA map, which is used for (among other things) the PIA's. In the Pegasus schematic, the two jumpers are shown "open" which indicates, or at least implies, that the default addressing for ROM3 and ROM2 is the $0xxx and $1xxx addressing, respectively. The use of the jumpers allows for these two ROMs to be re- mapped, either collectively or individually to the areas shown with this decoder. The MONITOR ROM, however, is always mapped to the $Fxxx area, as indicated. The reason for this is relatively straight-forward - the $Fxxx area contains the 6809 Interrupt Vectors. This is hard-coded into the 6809 CPU and cannot be changed, thus the MONITOR must ALWAYS be mapped into this area. The Interrupt Vectors are mapped as follows:

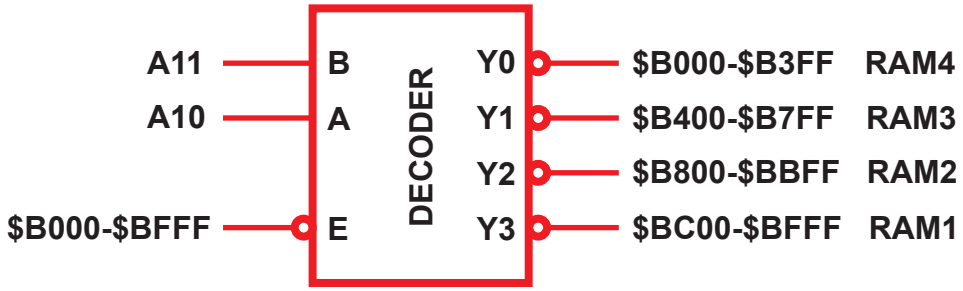
$FFF0-$FFF1 - Reserved (6809), Illegal Instruction (6309)
$FFF2-$FFF3 - SWI3
$FFF4-$FFF5 - SWI2
$FFF6-$FFF7 - FIRQ
$FFF8-$FFF9 - IRQ
$FFFA-$FFFB - SWI
$FFFC-$FFFD - NMI
$FFFE-$FFFF - RESET (and POWER ON)

The different Interrupts will fetch the corresponding Vector from the memory as listed above.

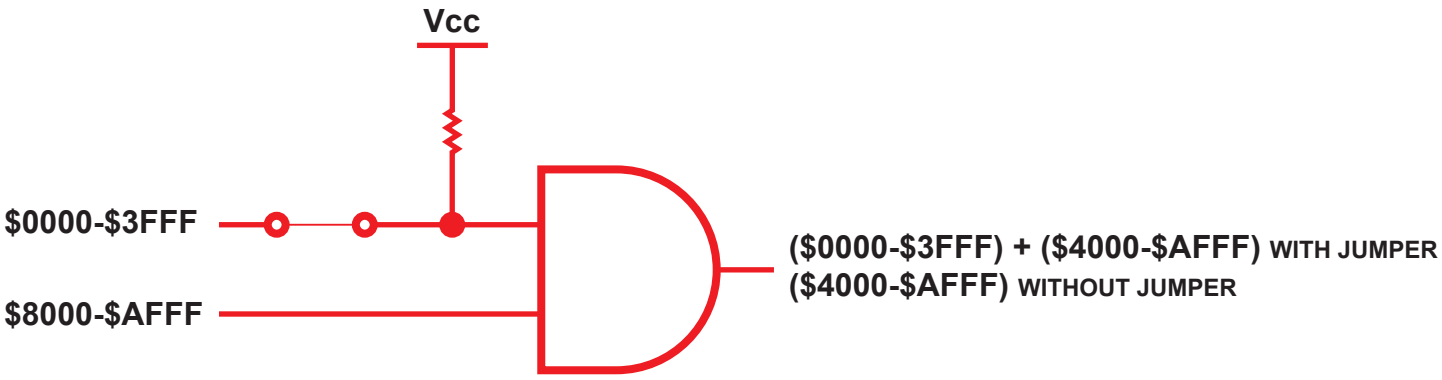


This section of the decoder takes the lower 2k of the 4k I/O area and decodes it into four 1/2k (512 byte) selects. The lowest 512 bytes are reserved, the next 512 bytes decode the Character RAM, and the two remaining 512 byte decodes are for the two PIA's. Note that the PIA's themselves DO NOT require the full 512 bytes assigned to them. As such, it is entirely possible to further decode these PIA sections into smaller blocks to allow for the addition of more PIA's or other hardware within this area.

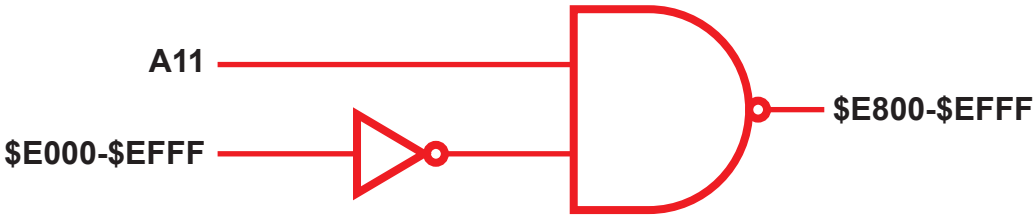
This section may seem a bit tricky. It takes the second and third 16k sections of the initial decode and combines them into a single decode MINUS the $B000-$BFFF area. Basically, the output of this section consists of two decodes - one for the $4000-$AFFF area, and the other only for the $B000-$BFFF area (which is the Pegasus onboard RAM area.) The $4000-$AFFF decode is combined with further logic to provide the decode for the 74LS245 buffer, which is intended for an SS-50 Expansion Bus.



This section of the decoder circuitry simply takes in the $Bxxx select from the previous section and splits it into four 1k selects for the Pegasus onboard RAM chips. If a single 4k RAM chip were used instead of 1k chips, then this decoder would not be needed.



This part of the decoder section also deals with the decoding of the 74LS245 buffer for the SS-50 bus. Note that the jumper is normally "OFF", which allows this section of the decoder to respond only to addresses from $4000-$AFFF. If the jumper is installed, then this section of the decoder will respond to all addresses in the range of $0000-$AFFF.



This section of the decoder decodes the External I/O area of the Pegasus ($E800-$EFFF).

This section of the decoder brings together ALL of the signals associated with the 74LS245 Data Buffer (for the SS-50 bus expansion) and creates the active low $\overline{\text{SEL}}$ signal that enables the buffer. This signal will be active on:

($0000..$3FFF [if related jumper is “ON”]) + ($4000..$AFFF) + ($E800..$EFFF)
OR
[if the jumper is "OFF"] ($4000..$AFFF) + ($E800..$EFFF)

That concludes the analysis of the Memory Map Decoder of the Pegasus. If something doesn't make sense, remember to have a look at "gate equivalencies" via your favorite search engine - it is necessary to understand the truth tables to basic gates and their equivalents in order to comprehend some of this. Mostly, though, it's quite simple. I'll answer any questions if you need to contact me:

sales@gimechip.com