Chapter 2

The Logic in the PLA

2.1 The Development

The logic needed to get the C64 running was programmed by Bob Yannes. "He just needed some glue logic to tie everything together," recalls James Redfield. According to Bil Herd, "[The original PLA terms] appear on an 82S100 worksheet photocopied out of the Signetics databook."

2.2 Reverse Engineering History

The memory maps in the Programmer's Reference Guide [PRG83] show all possible configurations as they are seen by the CPU, including Ultimax mode. However, some details cannot be taken from that book, especially memory configurations for VIC-II accesses.

Probably the earliest full reverse engineering of the logic programmed to the PLA was written by William Levak in 1986. It was published in The Transactor magazine [Lev86]. He read out a PLA, put the binary dump onto a disk and used a computer program to extract the logic table from it. Using this way he avoided transcription errors. He double-checked the results against two other PLAs. Unfortunately the article in The Transactor has some typesetting mistakes in the memory maps, although the logic table is correct and nicely optimized.

Some years later, in 1994, Jens Schönfeld - apparently not aware of William's work - captured the truth table again. He, Marko Mäkelä, Andreas Boose and others [PLA95] investigated that table and derived equations from it. Mark Smith took a programmer and read out an 82S100 in 1995. The results confirmed their work.

2.3 PLA Logic Revisions

Very early C64s contain a PLA which was labeled *REV2 7E17*. This version is said to have a different logic implemented. Unfortunately no binary or JEDEC dump was available for this document. The next version was *REV3 8411*. The four digit hex number is the checksum of the original JEDEC file most likely.

Different PLAs with different part numbers by different manufacturers found in C64s have been read out. The logic in all of them was identical with REV3, so this was the final revision.

2.4 How the PLA is Connected in the C64

To understand the logic in the PLA, first let's have a look how it is connected in the C64. All descriptions are based on the schematic #251138 from [Serv85], but apply for other C64 models too.

#CAS (I0)

The input line I0 is connected to the #CAS output of the VIC-II. In every Phi1 and Phi2 cycle this line is pulled down by the VIC-II to initiate a read or write access to DRAM. Depending on the other inputs, the PLA may propagate the #CAS pulse to the #CASRAM output or mask it out. Only if it is propagated to #CASRAM, DRAM is actually addressed by the memory access. When #CASRAM remains high in a cycle because it is disabled by the PLA logic, the DRAM access is rendered ineffective, or more precisely it turns to a RAS-only refresh cycle.

#LORAM, #HIRAM, #CHAREN (I1 to I3)

The lines #LORAM (I1), #HIRAM (I2) and #CHAREN (I3) are connected to the processor port of the CPU, better known as bits 0..2 of $00/$01 in the 6510/8500. After a reset these lines are all set to input mode by the CPU, which means that they are not driven. To make sure the have a sane value when the machine is started, they are pulled up by R43, R44 and R45. With all these values being 1, the C64 can start with KERNAL, I/O and BASIC banked in.

#VA14 (I4)

#VA14 (I4) is one of the additional video address lines for the VIC-II. The VIC-II has a 14 bit wide address bus to address 16 KiByte of memory, while the C64 has 64 KiByte of DRAM. To allow the VIC-II to address any of the four 16 KiByte chunks available in these 64 KiByte, two additional address lines called #VA14 and #VA15 are generated by programmable outputs of the CIA U2. They are fed to an extra address multiplexer U14. This is a 74LS258, which has inverted outputs.

The I/O ports are in input mode after the CIA has been reset. In this case these two lines are pulled high by internal pull-up devices in the CIA, which effectively selects bank 0 on startup. Note that these two video address bits do not appear on the global C64 address bus, they are only used for DRAM accesses. #VA14 is also connected to the PLA to influence the character set ROM mapping.

A15 to A12 (I5 to I8)

The address bus lines A15 to A12 (I5 to I8) are connected to the global address bus. They are driven by the CPU when AEC from the VIC-II and #DMA from the Expansion Port are both high. During VIC-II cycles, when AEC is low, they are pulled up by RP4. When they are pulled up by this resistor array only, it is possible to change them from the Expansion Port. These address lines are used by the PLA to control the memory mapping when AEC is high, i.e. during CPU cycles. However, they are also evaluated in the PLA in Ultimax mode when AEC is low, which makes an interesting trick possible, shown in section A.

BA (I9)

BA (I9) is set low by the VIC-II when it wants to halt the CPU to be able to use both half-cycles to get more memory bandwidth. When the VIC-II pulls down BA to take over the bus, there are three cycles left for the CPU. In these cycles up to three write accesses can take place, which is the maximum number of consecutive write accesses a 6510 does. However, the CPU stops its work before the first read access is started. Therefore there are up to three dummy read accesses on the bus until the VIC-II takes over control.

#AEC (I10)

#AEC (I10) is an inverted version of AEC. #AEC is high whenever the VIC-II is going to control the address bus. This is the case in every Phi1 cycle and in all full cycles when is BA low, after the CPU got three extra Phi2 cycles.

R/#W (I11)

The signal R/#W (I11) is high for read accesses and low for write accesses. Note that it is pulled up with resistor R51, because this line is not driven by the CPU when AEC or #DMA are low.

#EXROM, #GAME (I12, I13)

The two lines #EXROM and #GAME can be pulled down by cartridges to change the memory map of the C64, e.g., to map external ROM into the address space. When they are not pulled down from the cartridge port, resistors in RP4 pull them up.

VA13, VA12 (I14, I15)

The two address lines VA13 and VA12 are directly connected to the VIC-II. These lines are needed to address the 16k address space of the VIC-II. Note that #VA14 is from a completely different source, it is connected to a CIA output.

#CASRAM (F0)

The #CASRAM output is connected to the #CAS input of the DRAM chips. It is a gated version of the #CAS signal from the VIC-II. The VIC-II activates the #CAS signal in every half-cycle. Whenever a memory access has to address a different chip or port than the internal DRAM, the PLA disables the #CASRAM output, i.e. it remains high. This is also the case in the Ultimax memory configuration, where some address ranges simply have no memory selected at all.

The #CASRAM line requires the PLA to fullfill certain timing requirements, details can be found in section 3.3.

#BASIC, #KERNAL, #CHARROM (F1..F3)

The outputs #BASIC, #KERNAL, #CHARROM are connected to the chip select inputs of BASIC, KERNAL and character set ROMs. To address one of these chips, the appropriate line is pulled down.

2.5 How to Extract the Logic from a 82S100

The 82S100 PLA used in some early C64s can be read out to a JEDEC file with a programmer. This file contains a direct image of the fuse map programmed in the PLA.

This fuse map can be translated to product terms, which are AND combinations of inputs and inverted inputs. These product terms may be used in any of the eight sum terms, which are OR combinations. Finally, the fuse map describes which output shall be inverted. The meaning of the bits in the JEDEC file has been explained, e.g., in a mail by Segher Boessenkool [Boess12].

For this document a Python tool has been implemented, which converts the JEDEC file to a more readable logic table and to VHDL source code. The result is also used in the PLA replacement described in section 5.

2.6 Full Logic Table

The following table shows the AND and OR relations in the PLA in a very compact form. The symbol '*' means *AND*, '/' means *AND NOT* and '+' means *OR*. Human readable equations are listed in subsequent sections, the resulting memory maps can be found in appendix A.

```
------------------------------------------------------
      p 00000000001111111111222222222233
        01234567890123456789012345678901
Inputs
#CAS    ........*...................../*/
#LORAM  *...*......**..**..*...........
#HIRAM  ****.*...**..**....*.*.........
#CHAREN ...///...********..............
#VA14   ......**.......................
A15     ******..****************.///**..*
A14     /*****..***********///*.//*/*..*
A13     ***///../////////////**..*.*/../
A12     ...***..***********.....*.../..*
BA      .........*.*.*.*.*.............
#AEC    //////**/////////////////*......./
R/#W    ******../*/*/*/*/*/*.*........./
#EXROM  ../.././/.....////**/*/*******...
#GAME   **/**/*/.****//////./////////...
VA13    ......//..............*........
VA12    ......**..............*........
                                        Outputs
        ++++++++.++++++++++++++++++++.+.     #CASRAM
        +.............................. NOT #BASIC
        .++............................ NOT #KERNAL
        ...+++++....................... NOT #CHARROM
        ..............................+ NOT #GRW
        .........++++++++++............ NOT #IO
        ...................++........... NOT #ROML
        ......................+++........ NOT #ROMH
------------------------------------------------------
```

2.7 Original PLA Equations

This chapter shows all equations as extracted directly from a 82S100 and explains their meaning.

Note that the value of VADDR refers to a VIC-II address as mapped to the 64k memory space: Address bits 15 and 14 are the inverted values of #VA15 and #VA14, generated by the CIA U2, the remaining bits are generated by the VIC-II itself.

Product Terms

These product terms are AND combinations of the input signals and inverted input signals. They can be used in sum terms to determine the final output signals.

Product Term for #BASIC

If p0 is true, BASIC ROM is selected.

```
-- #LORAM = 1, #HIRAM = 1,
-- address $A000..$BFFF,
-- no VIC access, read, cartridge: none or 8k
p0 <= n_loram and n_hiram and
      a15 and not a14 and a13 and
      not n_aec and rd and n_game;
```

Product Terms for #KERNAL

If p1 or p2 are true, KERNAL ROM is selected.

```
-- #HIRAM = 1,
-- address $E000..$FFFF
-- no VIC access, read, cartridge: none or 8k
p1 <= n_hiram and
      a15 and a14 and a13 and
      not n_aec and rd and n_game;

-- #HIRAM = 1,
-- address $E000..$FFFF
-- no VIC access, read, cartridge: 16k
p2 <= n_hiram and
      a15 and a14 and a13 and not n_aec and
      rd and not n_exrom and not n_game;
```

Product Terms for #CHARROM

If one or more of p3 to p7 are true, CHARACTER SET ROM is selected.

```
-- #HIRAM = 1, #CHAREN = 0,
-- address $D000..$DFFF
-- no VIC access, read, cartridge: none or 8k
p3 <= n_hiram and not n_charen and
      a15 and a14 and not a13 and a12 and
      not n_aec and rd and n_game;

-- #LORAM = 1, #CHAREN = 0,
-- address $D000..$DFFF,
-- no VIC access, read, cartridge: none or 8k
p4 <= n_loram and not n_charen and
      a15 and a14 and not a13 and a12 and
      not n_aec and rd and n_game;

-- #HIRAM = 1, #CHAREN = 0,
```

```
-- address $D000..$DFFF,
-- no VIC access, read, cartridge: 16k
p5 <= n_hiram and not n_charen and
      a15 and a14 and not a13 and a12 and
      not n_aec and rd and not n_exrom and not n_game;

-- VADDR $1000..$1FFF or $9000..$9FFF
-- VIC-II access, cartridge: none or 8k
p6 <= n_va14 and not va13 and va12 and
      n_aec and n_game;

-- VADDR $1000..$1FFF or $9000..$9FFF
-- VIC-II access, cartridge: 16k
p7 <= n_va14 and not va13 and va12 and
      n_aec and not n_exrom and not n_game;
```

Unused Product Term p8

The term p8 is not used at all. It may be a remainder of an earlier design stage of the C64 prototypes. Note that this is the same as p31 with CAS inverted.

```
p8 <= n_cas and
      a15 and a14 and not a13 and a12 and
      not n_aec and not rd;
```

Product Terms for #IO

If one or more of p9 to p18 are true, an I/O chip or port is selected.

```
-- #HIRAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access bus available, read,
-- cartridge: none or 8k
p9 <= n_hiram and n_charen and
      a15 and a14 and not a13 and a12 and
      not n_aec and ba and rd and n_game;

-- #HIRAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, write, cartridge: none or 8k
p10 <= n_hiram and n_charen and
       a15 and a14 and not a13 and a12 and
       not n_aec and not rd and n_game;

-- #LORAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, bus available, read,
-- cartridge: none or 8k
p11 <= n_loram and n_charen and
       a15 and a14 and not a13 and a12 and
       not n_aec and ba and rd and n_game;

-- #LORAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, write, cartridge: none or 8k
p12 <= n_loram and n_charen and
       a15 and a14 and not a13 and a12 and
```

```
       not n_aec and not rd and n_game;

-- #HIRAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, bus available, read, cartridge: 16k
p13 <= n_hiram and n_charen and
       a15 and a14 and not a13 and a12 and
       not n_aec and ba and rd and
       not n_exrom and not n_game;

-- #HIRAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, write, cartridge: 16k
p14 <= n_hiram and n_charen and
       a15 and a14 and not a13 and a12 and
       not n_aec and not rd and
       not n_exrom and not n_game;

-- #LORAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, bus available, read, cartridge: 16k
p15 <= n_loram and n_charen and
       a15 and a14 and not a13 and a12 and
       not n_aec and ba and rd and
       not n_exrom and not n_game;

-- #LORAM = 1, #CHAREN = 1,
-- address $D000..$DFFF,
-- no VIC access, write, cartridge: 16k
p16 <= n_loram and n_charen and
       a15 and a14 and not a13 and a12 and
       not n_aec and not rd and
       not n_exrom and not n_game;

-- address $D000..$DFFF
-- no VIC access, bus available, read, cartridge: Ultimax
p17 <= a15 and a14 and not a13 and a12 and
       not n_aec and ba and rd and
       n_exrom and not n_game;

-- address $D000..$DFFF,
-- no VIC access, write, cartridge: Ultimax
p18 <= a15 and a14 and not a13 and a12 and
       not n_aec and not rd and n_exrom and not n_game;
```

Product Terms for #ROML

If p19 or p20 are true, the cartridge line ROML is selected.

```
-- #LORAM = 1, #HIRAM = 1,
-- address $8000..$9FFF
-- no VIC access, read, cartridge: 8k or 16k
p19 <= n_loram and n_hiram and
       a15 and not a14 and not a13 and
       not n_aec and rd and not n_exrom;

-- address $8000..$9FFF
```

```
-- no VIC access cartridge: Ultimax
p20 <= a15 and not a14 and not a13 and
       not n_aec and n_exrom and not n_game;
```

Product Terms for #ROMH

If one or more of p21 to p23 are true, the cartridge line ROMH is selected.

```
-- #HIRAM = 1,
-- address $A000..$BFFF,
-- no VIC access, read, cartridge: 16k
p21 <= n_hiram and
       a15 and not a14 and a13 and
       not n_aec and rd and not n_exrom and not n_game;

-- address $E000..$FFFF,
-- no VIC access cartridge: Ultimax
p22 <= a15 and a14 and a13 and
       not n_aec and n_exrom and not n_game;

-- VADDR $3000..$3FFF, $7000..$7FFF, $B000..$BFFF or
--       $E000..$EFFF,
-- VIC-II access, cartridge: Ultimax
p23 <= va13 and va12 and
       n_aec and n_exrom and not n_game;
```

Additional Product Terms for #CASRAM

The DRAM of the C64 must be disabled whenever another device is selected. Therefore all product terms above appear in the #CASRAM term. Exceptions are the unused terms and #GRW, which is not a chip select signal. However, in Ultimax mode most of the DRAM is hidden permanently to emulate the 4 KiByte contained in an original Commodore MAX Machine. To hide the remaining DRAM, the terms p24 to p28 are used.

```
-- address $1000..$1FFF, $3000..$3FFF,
-- cartridge: Ultimax
p24 <= not a15 and not a14 and a12 and
       n_exrom and not n_game;

-- address $2000..$3FFF,
-- cartridge: Ultimax
p25 <= not a15 and not a14 and a13 and
       n_exrom and not n_game;

-- address $4000..$7FFF,
-- cartridge: Ultimax
p26 <= not a15 and a14 and
       n_exrom and not n_game;

-- address $A000..$BFFF,
-- cartridge: Ultimax
p27 <= a15 and not a14 and a13 and
       n_exrom and not n_game;

-- address $C000..$CFFF,
-- cartridge: Ultimax
```

```
p28 <= a15 and a14 and not a13 and not a12 and
       n_exrom and not n_game;
```

Unused Product Term p29

Term p29 is unused. Note that this is the same as p30 with CAS inverted.

```
p29 <= not n_cas;
```

Product Term to Forward #CAS to #CASRAM

p30 is used to put #CASRAM always high when #CAS is high. This completes the gate mechanism for #CAS.

```
p30 <= n_cas;
```

Product Term for #GRW

The C64 makes use of static RAM for the color memory. Typical SRAM parts like the HM472114 ([Hita2114]) need their address lines to be set up for a certain time before they get their chip select and write enable signals. Because of the bus multiplex mechanism controled by #AEC this is not the case in the C64.

When the color SRAM may be written, a special #GRW is generated for it. The term makes use of #CAS, because this input is only active when the address bus has a stable state. Note that also the I/O decoding has to match to enable an actual write access to the color RAM.

```
-- #CAS low,
-- address $D000..$DFFF,
-- no VIC access, write
p31 <= not n_cas
       and a15 and a14 and not a13 and a12 and
       not n_aec and not rd;
```

Sum Terms

```
-- No RAM whenever BASIC, KERNAL, CHARROM, IO,
-- ROML or ROMH are accessed or
-- any area with RAM disabled in Ultimax mode or
-- when there is no CAS signal from the VIC-II
n_casram <= p0 or p1 or p2 or
            p3 or p4 or p5 or p6 or p7 or
            p9 or p10 or p11 or p12 or p13 or
            p14 or p15 or p16 or p17 or p18 or
            p19 or p20 or p21 or p22 or p23 or
            p24 or p25 or p26 or p27 or p28 or p30;

-- Low for BASIC ROM read
n_basic <= not p0;

-- Low for KERNAL ROM read
n_kernal <= not (p1 or p2);

-- Low for CHARACTER SET ROM read
n_charrom <= not (p3 or p4 or p5 or p6 or p7);
```

```
-- Clean write pulse for color RAM
n_grw <= not p31;

-- Low for I/O chips or ports read or write
n_io <= not (p9 or p10 or p11 or p12 or p13 or p14 or
             p15 or p16 or p17 or p18);

-- Low for cartridge ROML read or write
n_roml <= not (p19 or p20);

-- Low for cartridge ROMH read or write
n_romh <= not (p21 or p22 or p23);
```