Replacement of some old prom by a GAL

Application to a Thaler MPS-65 board

When a prom contains a program, it must only be replaced by a similar prom or adapter using a flash memory. But when the prom contains some logic to produce chip selects there is a good chance that this prom can be replaced by a GAL

We will use the fact that the pins of the GAL can be mapped almost as the one of some proms except for the power supply.

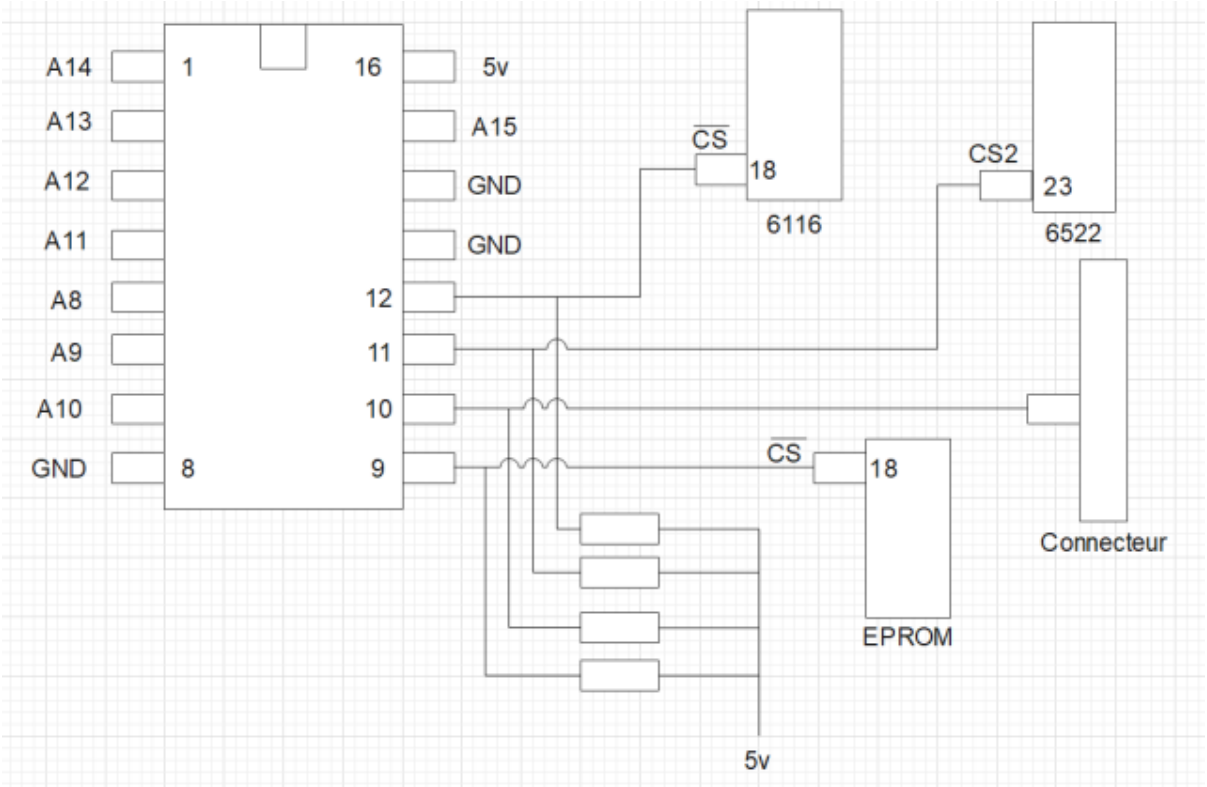
The best is to try to find a dump of the prom on internet…

But if it's not working, we have to rebuild it

First, we must find the schematics or rebuild a small schematic.

With the connection between the prom and the chips

Here these schematics have been built by reverse engineering of a board.



With this schematic we can determine the role of each output pin

The connection of the pin A15.A8 have been recovered by a examining a not working prom dump and the partial schematics of a similar board. (CT-65)

On a a simple board and the cpu a 6502 we can deduce that the eprom is at the top of the memory, and the eprom capacity can be obtained by the eprom

Type. For the ram if there is only one chip it is not difficult, on a 6502 it must be located on the bottom of memory… For the peripherals it's more difficult, without any data found on the web, the only solution is to disassemble the code in the eprom and analyse it

Ressources found on the web:

Partial manual and schematics of CT-65 board

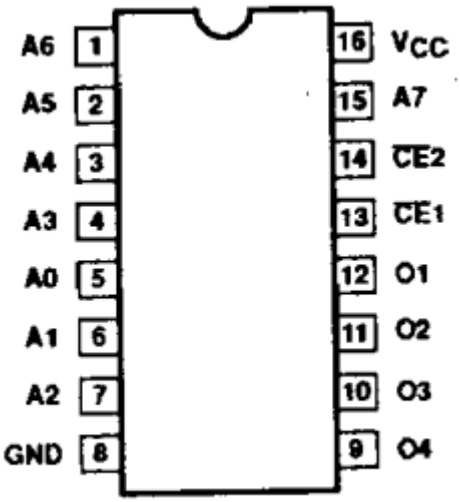
(Machine party similar to the MDS-65 board)

Dump of the prom (gave some idea but was not working)

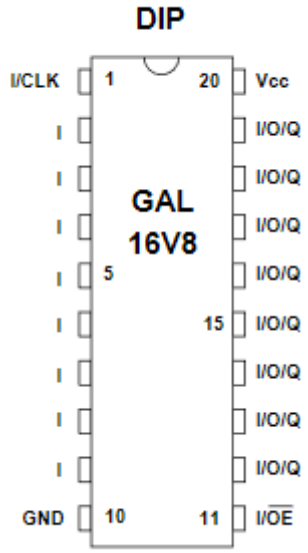
Photo of the MDS-65 machine

In our case, the prom is an 82S126, 256x4 prom with output open collector.

From the 82s126 datasheet we can get the 82s126 pinout



From the GAL 16v8D datasheet we can get the 16v8D pinout

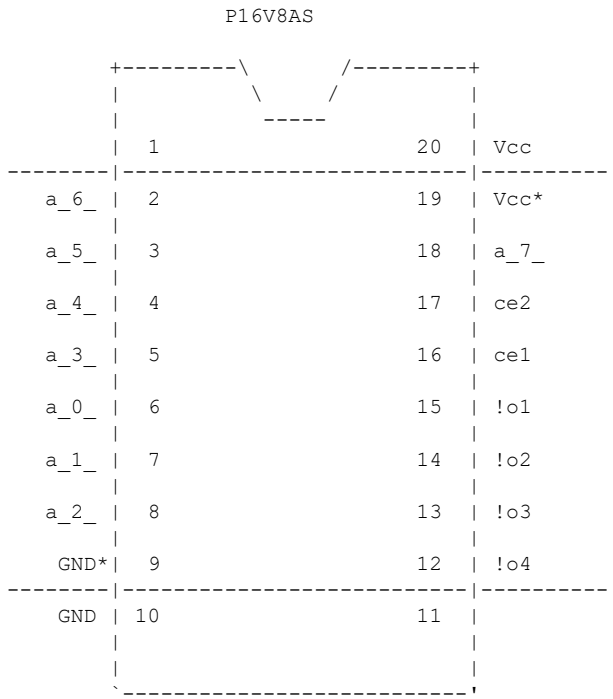


We can see that:

pin 2..9 of the GAL are input and pin 1..7 of the prom are input

Pin 12..19 of the GAL can be input/output and pin 10..15 of the prom are either input or output

So the idea is to map pin to pin the prom and the gal this way:



There is still a problem the pins 1,10, 11, 20 of the GAL are not connected.

If we place the GAL directly on the prom socket

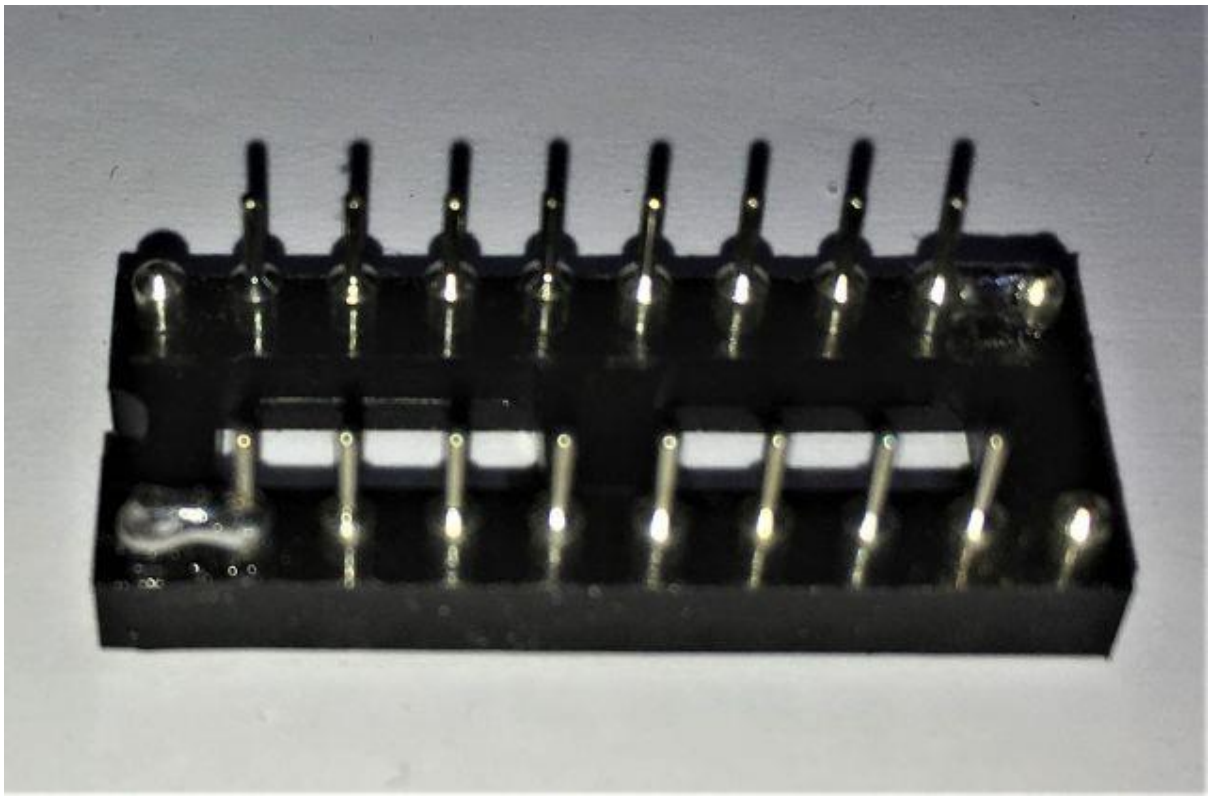
So a small adapter is needed

To make the adapter the easiest is to take a 20-pin socket.

And make a bridge of solder between pin 9 and pin 10 to connect GND

And make a bridge of solder between pin 19 and pin 20 to connect VCC

It’s not the best design but it works…



About open collector output of the prom, you normally have resistor on these outputs… but the GAL is not open collector, so you have to remove them or at least disconnect one side from the GAL…

For the programming of the GAL

I used ISP Lever Classic and vhdl code.

There is a problem with this product, in the past there was a free licence available. But now you have to pay for a licences

The best would be to use alternative solution such as Wincupl or galasm

With ISPLever Classic, create a project, and select GAL 16V8D

(not you have to check “obsolete product” checkbox to access the GAL list)

vhdl file for the Thaler MDS-65

the GAL pins must match the prom pins

as CE1 et CE2 are tied to GND (always valid)

in the GAL they are just ignored

O1 of the prom is connected to the 2K ram chip at the bottom of memory

```
o1 <= '0' when (A >= x"00") and (A <= x"07") else '1';
```

O2 of the prom is connected to the via at $A000

```
o2 <= '0' when (A = x"A0") else '1';
```

O3 of the prom is not used, so it is set to high

```
o3 <= '1';
```

O4 of the prom is connected to 4K eprom at the top of memory

```
o4 <= '0' when (A >= x"F0") and (A <= x"FF") else '1';
```

```
library ieee;

 use ieee.std_logic_1164.all;
 use ieee.std_logic_arith.all;
 use ieee.std_logic_unsigned.all;

-- first we defined the signals used
entity md65 is
      PORT (
            a        : in   std_logic_vector(7 downto 0);
            ce2      : in   std_logic;
            ce1      : in   std_logic;
            o1       : out  std_logic;
            o2       : out  std_logic;
            o3       : out  std_logic;
            o4       : out  std_logic
      );

-- then we associate the signals with a pin number
attribute LOC : string;
-- P1  not connected
attribute LOC of a         : signal is "P18,P2,P3,P4,P5,P8,P7,P6";
-- P9  connected to P10 via strap
-- P10 GND
-- P11 not connected
attribute LOC of ce2       : signal is "P17";
attribute LOC of ce1       : signal is "P16";
attribute LOC of o1        : signal is "P15";
attribute LOC of o2        : signal is "P14";
attribute LOC of o3        : signal is "P13";
attribute LOC of o4        : signal is "P12";
-- P19 connected to P20 via strap
-- P20 VCC

end;

architecture behavioral of md65 is
begin
      o1 <= '0' when (A >= x"00") and (A <= x"07") else '1';  -- ram
      o2 <= '0' when (A = x"A0")                   else '1';  -- via
      o3 <= '1';                                              -- not used
      o4 <= '0' when (A >= x"F0") and (A <= x"FF") else '1';  -- eprom
end behavioral;
```

Once this file is compiled mapping verified by displaying the “chip report”

It is possible to burn a 16V8D GAL chip with the Jedec file

Plug the GAL on the board and power it and Press R

ispLEVER Classic 2.1.00.02.49.20 - Device Utilization Chart

Module : 'md65'

Input files:

ABEL PLA file : untitled.tt3
Device library : P16V8AS.dev

Output files:

Report file : untitled.rpt
Programmer load file : untitled.jed

---

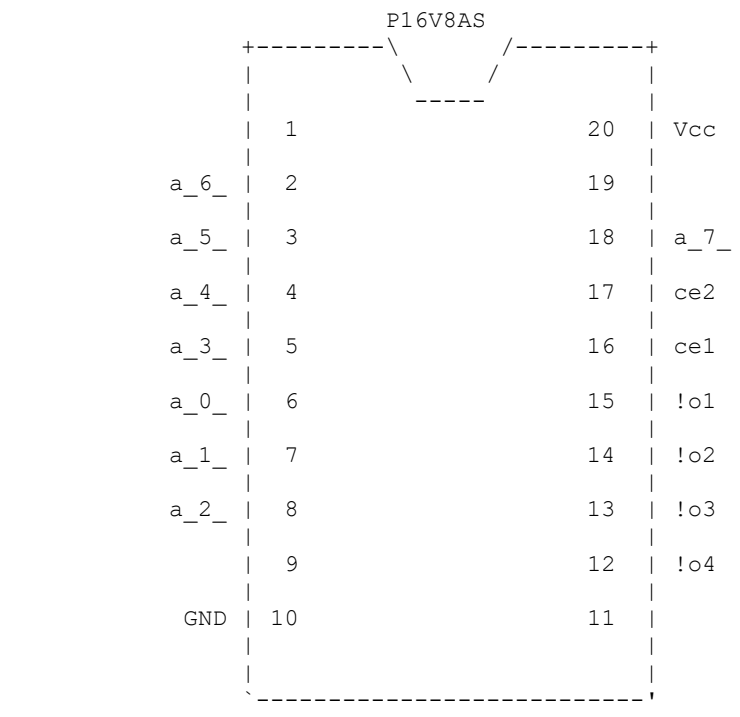
ispLEVER Classic 2.1.00.02.49.20 - Device Utilization Chart

P16V8AS Programmed Logic:

```
o1     = !(  !a_7_ & !a_6_ & !a_5_ & !a_4_ & !a_3_ );
o2     = !(  a_7_ & !a_6_ & a_5_ & !a_4_ & !a_3_ & !a_2_ & !a_1_ & !a_0_ );
o3     = !(0);
o4     = !(  a_7_ & a_6_ & a_5_ & a_4_ );
```


ispLEVER Classic 2.1.00.02.49.20 - Device Utilization Chart

P16V8AS Chip Diagram:



SIGNATURE: N/A

ispLEVER Classic 2.1.00.02.49.20 - Device Utilization Chart

P16V8AS Resource Allocations:

Device Resources	Resource Available	Design Requirement	Unused
Input Pins:			
Input:	10	7	3 (30 %)
Output Pins:			
In/Out:	6	4	2 (33 %)
Output:	2	1	1 (50 %)
Buried Nodes:			
Input Reg:	-	-	-
Pin Reg:	-	-	-
Buried Reg:	-	-	-

ispLEVER Classic 2.1.00.02.49.20 - Device Utilization Chart

P16V8AS Product Terms Distribution:

Signal Name	Pin Assigned	Terms Used	Terms Max	Terms Unused
o1	15	1	8	7
o2	14	1	8	7
o3	13	0	8	8
o4	12	1	8	7

==== List of Inputs/Feedbacks ====

Signal Name	Pin	Pin Type
a_7_	18	BIDIR
a_6_	2	INPUT
a_5_	3	INPUT
a_4_	4	INPUT
a_3_	5	INPUT
a_2_	8	INPUT
a_1_	7	INPUT
a_0_	6	INPUT

ispLEVER Classic 2.1.00.02.49.20 - Device Utilization Chart

P16V8AS Unused Resources:

Pin	Pin	Product	Flip-flop

Number	Type	Terms	Type
1	INPUT	-	-
9	INPUT	-	-
11	INPUT	-	-
16	OUTPUT	NORMAL 8	-
17	BIDIR	NORMAL 8	-
19	BIDIR	NORMAL 8	-

Thaler MDS-65 board working with it’s prom replaced by a 16V8D GAL