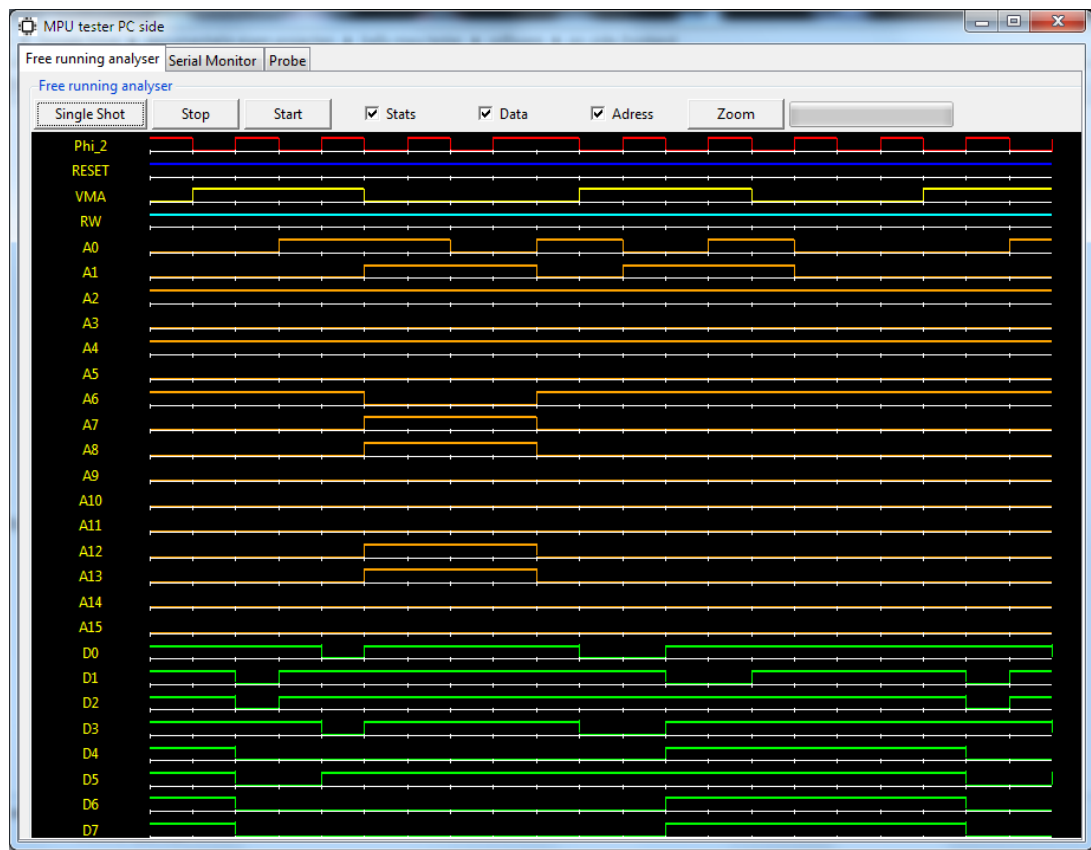


# André's MPU 35 tester



Author: André Boot

© Andre Boot 2017

## Contents

<b>1. INTRODUCTION .....</b>	<b>2</b>
<b>2. REQUIRED HARDWARE.....</b>	<b>4</b>
<b>3. BUILDING THE MPU TESTER .....</b>	<b>5</b>
3.1 CABLE LAYOUT/SCHEMATIC .....	6
<b>4. PROGRAMMING ANDRÉ MPU TESTER.....</b>	<b>10</b>
4.1 INSTALLING USB DRIVERS .....	10
4.2 PROGRAMMING THE MEGA2560 .....	10
<b>5. USING MPU TESTER STAND-ALONE .....</b>	<b>10</b>
5.1 INFO ABOUT KEYS .....	11
5.2 TEST 2 THRU 8: CONTINUITY TEST OF U2, U6, U7, U8, U9, U10,U11 .....	11
5.3 TEST 9: LOGIC ANALYSER .....	14
5.4 TEST 10: PROBE READOUT .....	16
5.5 TEST 11: ADDRESS AND DATALINES .....	17
5.6 TEST12: CONNECTOR J1 .....	17
5.7 TEST13: CONNECTOR J2 .....	19
5.8 TEST14: CONNECTOR J3 .....	19
5.9 TEST15: CONNECTOR J4 .....	19
5.10 TEST16: CONNECTOR J5 .....	19
5.11 TEST17: READ PHI .....	20
5.12 TEST 18: READ RESET .....	20
5.13 TEST 19: 5101 TEST .....	20
5.14 TEST 20: 6810 TEST .....	20
5.15 TEST 21: LED ON OFF.....	20
5.16 TEST 22: U14 CD4049.....	20
5.17 TEST 23: U15 MC3459L=7437 .....	20
5.18 TEST 24 THROUGH 27 (U16=9602, U17=74L00, U18= 4049, U19= 4011).....	21
5.19 TEST 28 DISPLAY INTERRUPT TEST.....	21
<b>6. USING A PC TO CONTROL ANDRÉ'S MPU TESTER.....</b>	<b>22</b>
6.1 PC SOFTWARE TO CONTROL THE MPU TESTER .....	22
6.1.1 <i>Using andre_mpu_gui.exe to control the tester. ....</i>	<i>22</i>
6.1.2 <i>Other software to control the tester .....</i>	<i>24</i>
6.2 TURNING ON MPU TESTER .....	26
6.3 LOCKED MPU (LED ALWAYS ON) TIPS .....	27
6.4 ADVANCED CONTINUITY CHECK.....	27
6.5 MEMORY DISPLAY OF EPROM U6 .....	29
6.6 TEST 84: TEST BOARD WITHOUTH EPROMS, LOAD TEST SOFTWARE INTO 6810.....	30
6.7 LOGIC ANALYZER FUNCTION CHECKING STARTUP .....	31
6.8 IC TESTS .....	33
6.9 TEST 83: DISPLAY MEMORY .....	33
<b>7. APPENDIX.....</b>	<b>33</b>
7.1 TESTING THE J5-CABLE .....	33
7.2 J4 CONNECTOR INTERFACE .....	35
7.3 AVR_DUDE INFO. ....	35
7.4 ARDUINO CODE SNIPPET DECLARING I/O .....	35

## 1. Introduction

André's MPU 35 Boot tester is an advanced continuity tester to find faults in locked up bally MPU's of type \_35 and \_17

It is made of commodity stuff like: resistors, dupont wires , arduino mega, and a "led and key" module which can be bought from various sources.

Functions of the tester:

- Do advanced continuity check of :
  - The mc6800 socket
  - The two pia sockets
  - The mc6810 socket
  - The 5101 socket
  - The U2 eprom socket
  - The U6 eprom socket
- Simple logic analyzer function to check the startup via a (time and state with mnemonics ). Sampling rates between 100ns and 300ns.
- See status of reset.
- Test correct operation of and clock generation  $\phi 2$
- Test ram chips U7 (MC6810), U8 (5101)
- Test display interrupt generator (U12)
- Check the various logic gates IC's:
  - U14 : CD4049
  - U15 : MC3459L=7437
  - U16 : 9602
  - U17 : 74L00N
  - U18 : CD4049
  - U19 : CD 4011
- Hardware Self-test J5 (check correct wiring of the tester is correct)
- Check for stuck data bits

## André's MPU 35 tester

### Background :

It was all created in 2017.

Why did I make this?

I have these bad corroded mpu's lying around and found it difficult to check if there was a lack of continuity between the pcb lines and IC sockets. There are very helpful documents, but counting pins of sockets and looking at the schematics and hoping the multimeter beeps to find continuity is quite difficult for me. So it began and the continuity tester worked, then I expanded it to include the pins that were driven by logic gates. And added a bunch more of code just for fun. During the build of the software I bricked only one Bally MPU 35 board (up to now) and several ic's en zero arduino's . I did need to buy a logic analyser to see if the my own written analyser gave correct results. Lastly I forgot about the project. Dug it up again. Then I wrote this document so I remember how to use it in future. Maybe it can be useful for others too.

Use at your own risk. Better safe than sorry, better not build it and not use it at all.

If you really like this you can buy me a cup of coffee:  
<https://www.buymeacoffee.com/andreboot>

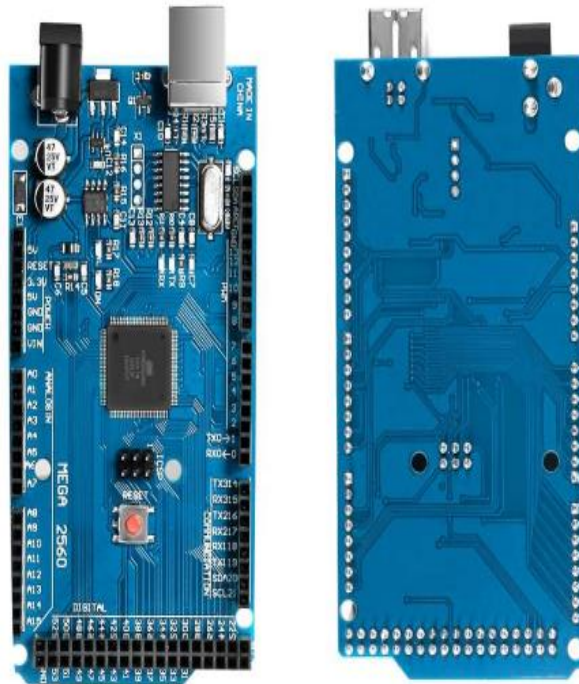
Ps.

*Sharing this software/manual/hardware for free is encouraged  
You are not allowed to sell this copyrighted document and copyrighted software  
except with approval of the author. If the tester/this document is sold for money  
without approval you agree to pay 50 cups of coffee to the author for each sold tester  
<https://www.buymeacoffee.com/andreboot>.*

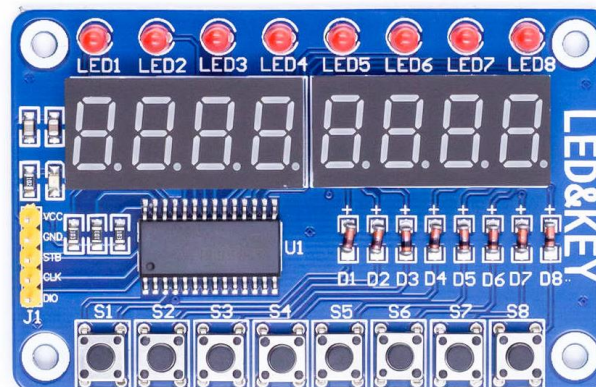
This documents highlights some of the most useful test.

## 2. Required hardware

1. To build and use the bally boot tester you will need:
  - Arduino mega



- A led and key thingy

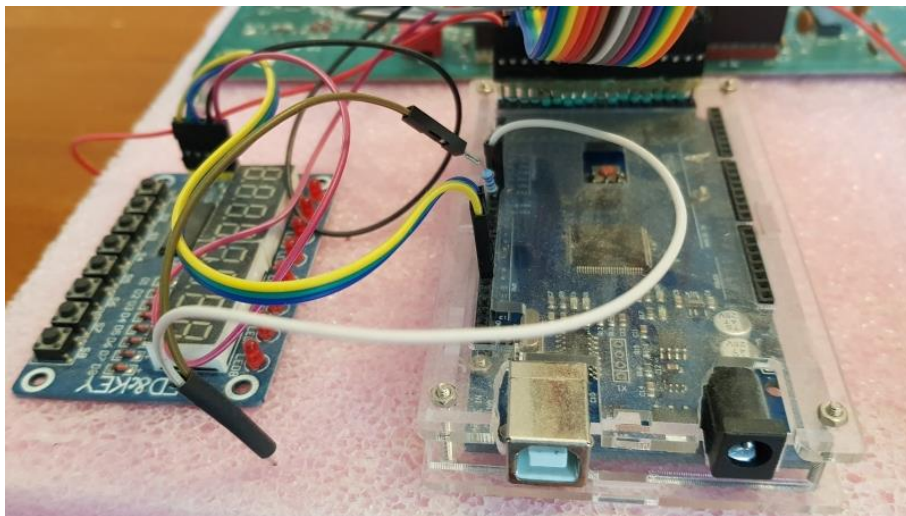


- 40 to 80 dupont jumpers (depending how you make the cable)
- A lot of resistors:
  - 33 x 6.2 kohm
  - 8 x 300 ohm
  - 3 x 340 ohm
  - 1x10 kohm
  - Usb cable
- Nice:
  - Some ic grabbers

## 3. Building the MPU tester

To build the tester only a lot of resistors and wires to connect to the modules and the MPU connector. The layouts/schematics/build plans are shown below.

In the image below version of the tester is shown, using the resistors as way to connect to connectors.

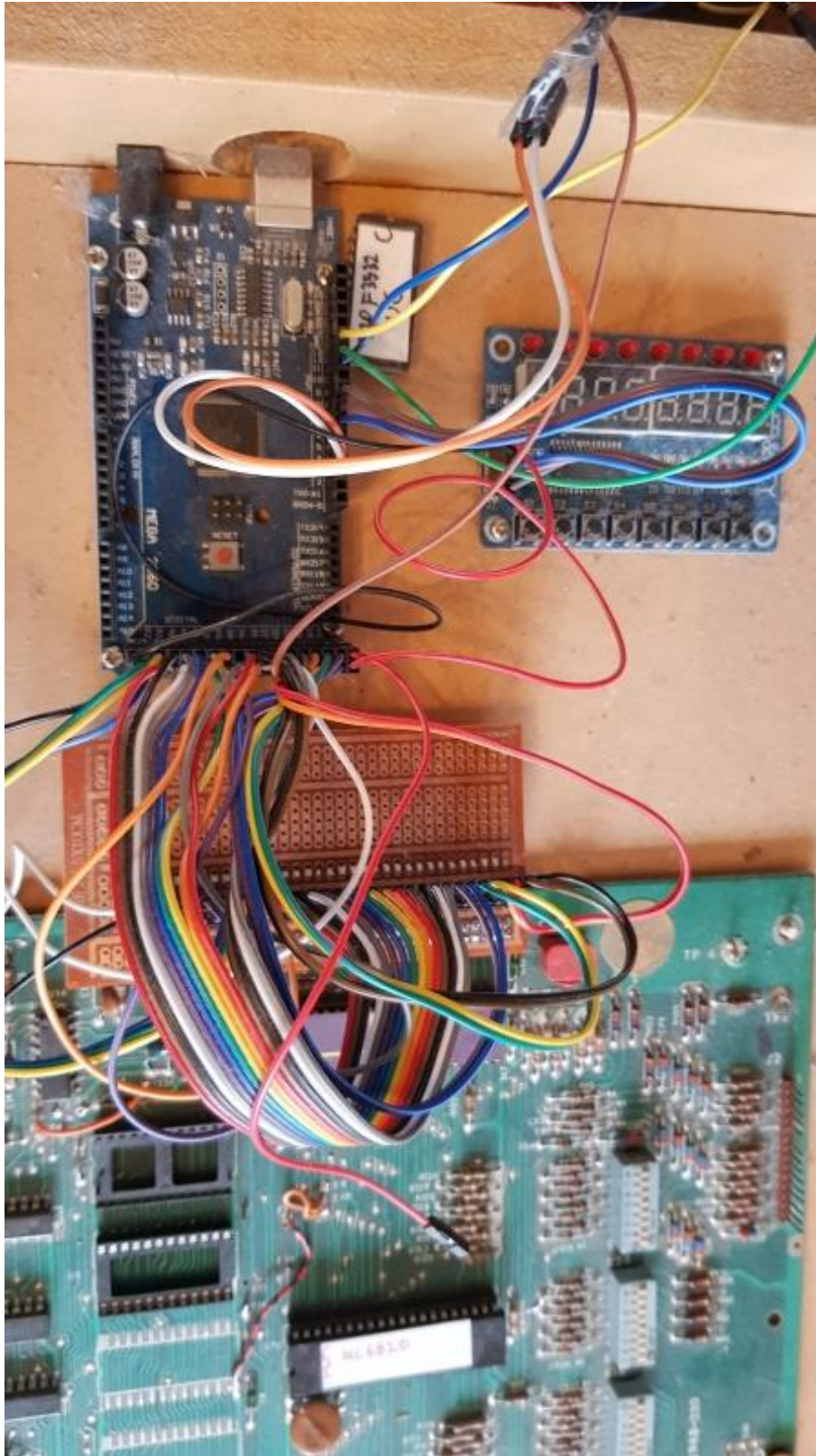


The Dupont wires can be used to connect to J5. Make sure to clean and sand J5 connector very well.



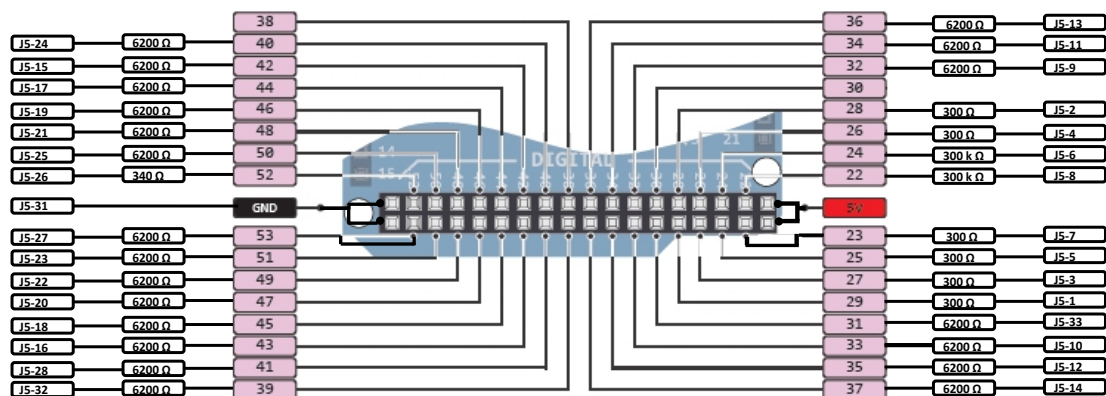
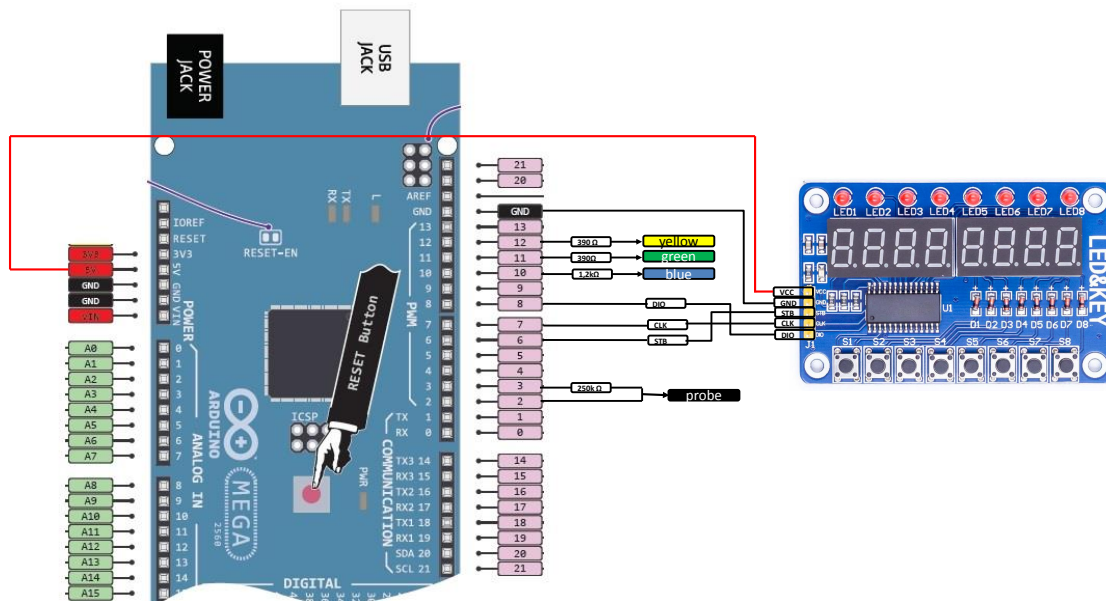
Or use a solderbreadboard to create the a messy tester. I did not create a special PCB because I wanted it to be created from commodity components.





### 3.1 Cable layout/schematic

# André's MPU 35 tester



Summary of J5-Arduino cable resistor values:

D0-D7 = 300 ohm  
VMA = 340 ohm  
Others = 6.2 Kohm

The following table lists connections and the resistor values

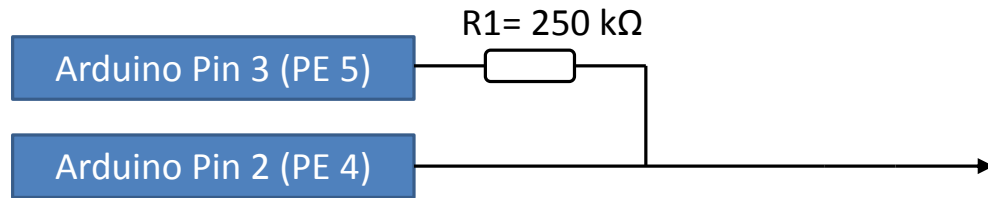
Bally Connector	Name	Resistor value in ohm ( between J5 and arduino pin)	Arduino connector
J5_1	D_7	300	=29;
J5_2	D_6	300	=28;
J5_3	D_5	300	=27;
J5_4	D_4	300	=26;



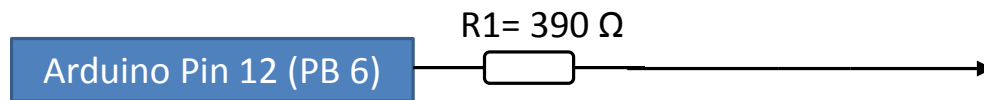
# André's MPU 35 tester

J5_5	D_3	300	=25;
J5_6	D_2	300	=24;
J5_7	D_1	300	=23;
J5_8	D_0	300	=22;
J5_9	A_13	6200	=32;
J5_10	A_12	6200	=33;
J5_11	A_11	6200	=34;
J5_12	A_10	6200	=35;
J5_13	A_9	6200	=36;
J5_14	A_8	6200	=37;
J5_15	A_7	6200	42;
J5_16	A_6	6200	=43;
J5_17	A_5	6200	=44;
J5_18	A_4	6200	=45;
J5_19	A_3	6200	=46;
J5_20	A_2	6200	=47;
J5_21	A_1	6200	=48;
J5_22	A_0	6200	=49;
J5_23	r_w	6200	=51;
J5_24	HLT	6200	=40;
J5_25	nRes	6200	=50;
J5_26	VMA	340	=52;
J5_27	Phi_2	6200	=53;
J5_28	Ext_mem	6200	=41;
J5_29	key //key	INFINITY OHM	not connected
J5_30	5V (VCC) //+5V	INFINITY OHM	not connected
J5_31	0V (GND) //0V	ZERO OHM	GND PIN (left of pin 52 vma)
J5_32	U11_pin18	6200	=39;
J5_33	A_14	6200	=31;

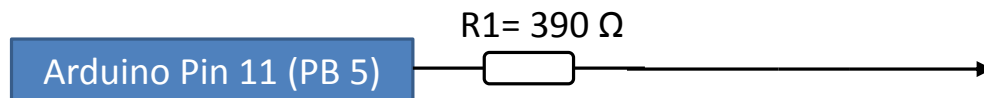
## Probe schematic



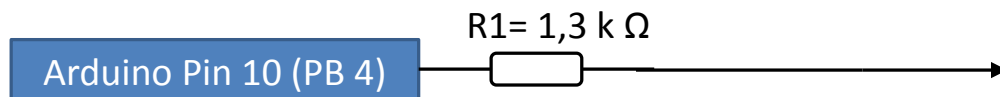
## Yellow Line schematic



## Green Line schematic



## Blue Line schematic



## Led and key device connection table

Led and key module	Resistor value in ohm ( led and key module and arduino pin)	Arduino connector number
DIO (Data)	0 (wire)	8
CLK (Clock)	0 (wire)	7
STB (Strobe)	0 (wire)	6
GND	0 (wire)	GND
VCC	0 (wire)	+5V

Yellow, blue and green-line are not really needed for 99% of the testing and repairing.

## 4. Programming André MPU tester.

This is the hard part and I am only giving several pointers. The Arduino needs to be programmed. This is written rather cryptic in the section below.

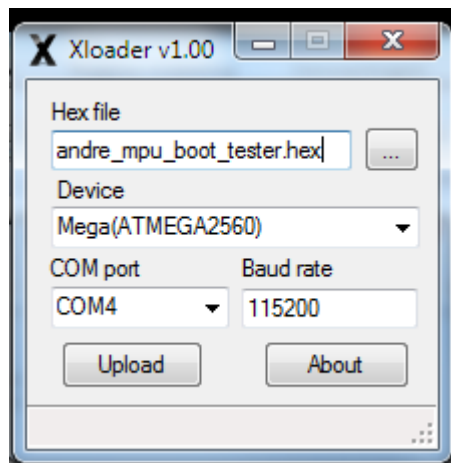
### 4.1 Installing USB drivers

Some Chinese Arduino clones are based on the ch340. To get it working you have to download a driver. Search for the driver via google and keyword CH340 drivers or try: <https://github.com/HobbyComponents/CH340-Drivers/>

### 4.2 Programming the mega2560

First install the ch340 usb driver if necessary.  
After installing the drivers. Use <https://www.hobbytronics.co.uk/arduino-xloader>

To upload the hex `andre_mpu_tester_v173.ino.hex`  
to the mega2560



Or use the commandline command at the folder where xloader is unpacked.

```
avrdude -Cavrdude.conf -v -patmega2560 -cwiring -PCOM4 -b115200 -D -Uflash:w: andre_mpu_tester_v173.ino.hex:i
```

## 5. Using MPU tester stand-alone

Using the MPU tester stand-alone not all tests can be accessed. Only 29 tests can be selected. These tests should be the most useful tests to repair boards.

Important For test 2,3,4,5,6,7,8, and 16 Make sure to remove all IC's: These are all roms, U2,U6,U7,U8,U9,U10,U11

**IMPORTANT: Remove all IC's: 6800, 2x6821,5101, 6810, Eproms and proms.**

**IMPORTANT: Remove all IC's: 6800, 2x6821,5101, 6810, Eproms and proms.**

**IMPORTANT: Remove all IC's: 6800, 2x6821,5101, 6810, Eproms and proms.**

After powering up the arduino and the MPU board, the led and key module will display a welcome message.

After that it will display the test selected. There are 29 tests which can be selected.

Using S8 on the led and key module advances to another test.  
Using S7 on the led and key module steps back.

### 5.1 Info about keys

Executing test 1 gives info about the test keys:  
S1 executes/starts a test  
S8 next test  
S7 previous test  
S6 for some tests this ends the test.

### 5.2 Test 2 thru 8: Continuity test of U2, U6, U7, U8, U9, U10,U11

Each pin of the selected IC can be checked to check whether there is continuity and correct function.

Using S1 on the led and key module executes the test selected.

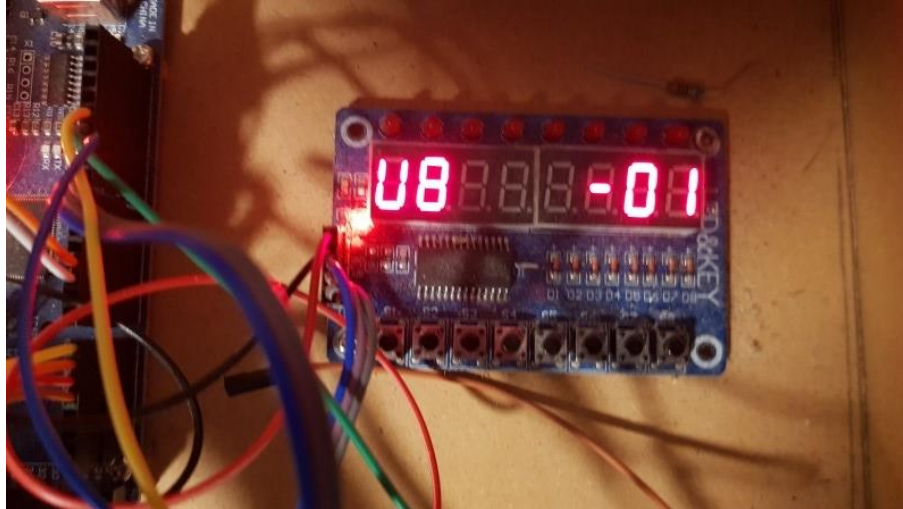
For instance if U8 socket needs to be checked:  
Press S8 until "2. U8" is visible on the led and key module  
Press S1 to start the test.

It should display U8 01.

If a pin is not tested. The led will display - - - before the pin number. For instance U11 - - - 02

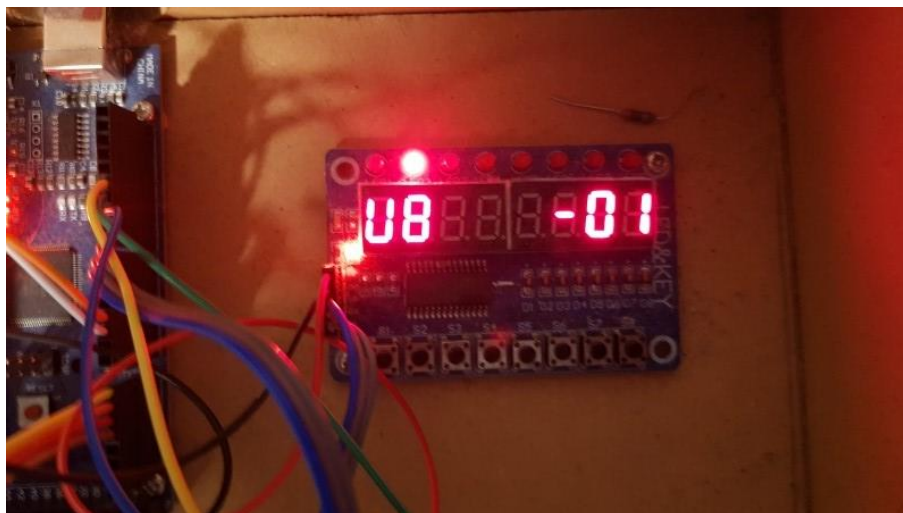
## André's MPU 35 tester

It should show you the following on the led and key module:



It displays which chip: U8 and which pin is activated.

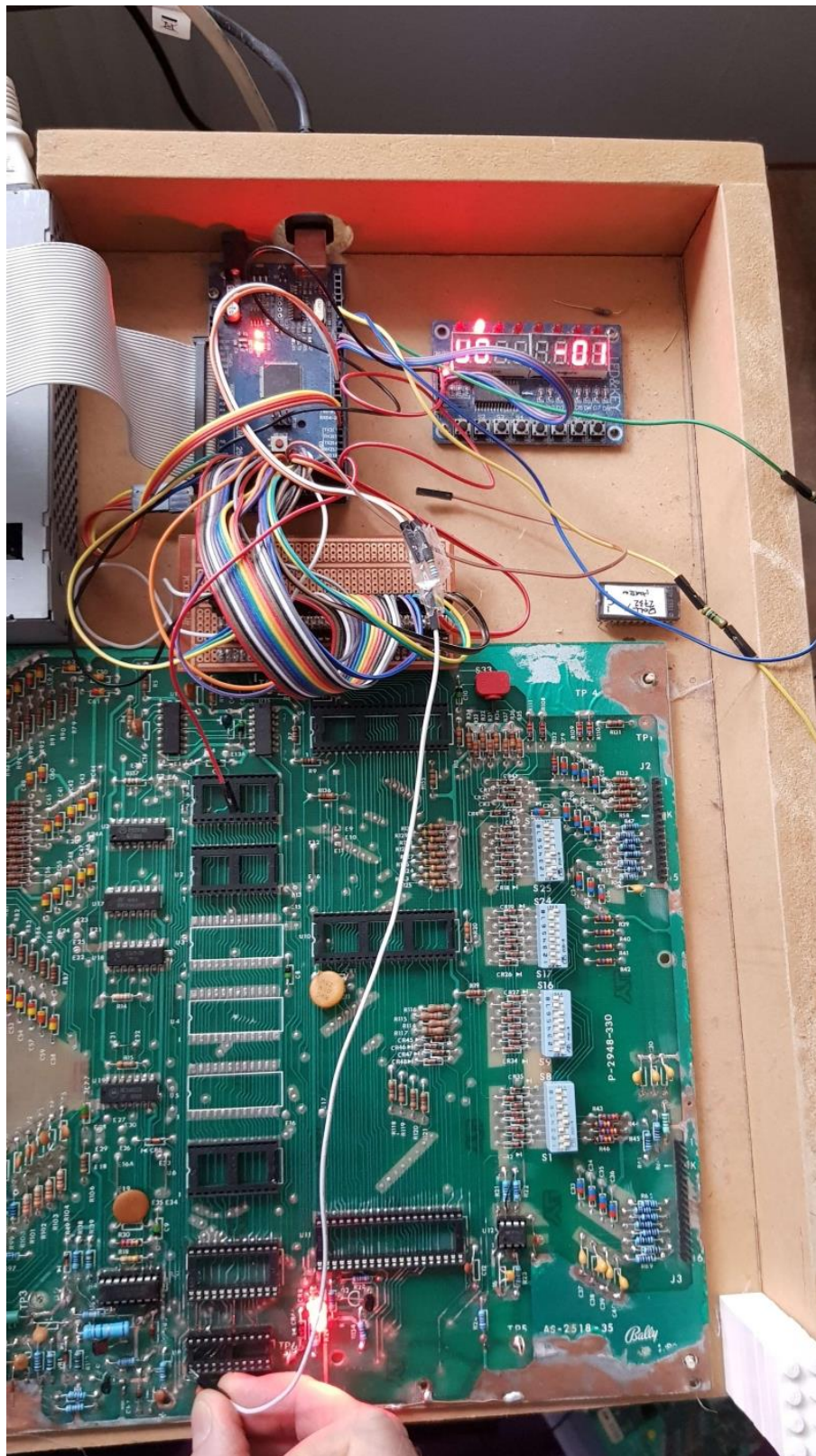
Take the probe (in my case the white wire). Connect it to pin 1. The 2nd led of the "Led and key" will be lite up, if there is continuity to this pin. Making probe contact to pin 1 will light up led 2 again.



And lifting the probe in the air will turn off led 2. If this is not the case there is a problem connecting to Pin 1 of U2.



## André's MPU 35 tester



To check if the pins of the socket work. Start at pin 1 with the probe. The led should go on. Put the probe at pin 2. Press S8 on the "led and key" module. Normally the led would go off. The led and key will display the next pin its

## André's MPU 35 tester

actuating. If you put the probe at pin 2, the led should go on, now press S8, the led should go out, now put probe at pin 3, led should go on , etc. etc.

Press S8 for the next pin until last pin is reached. If you press S8 again you will return to the menu on the led and key.

### 5.3 Test 9: Logic Analyser

All chips should be present for this test.

Before this test reset should be made low (remove 12V from the MPU) then select the test, make reset high again.

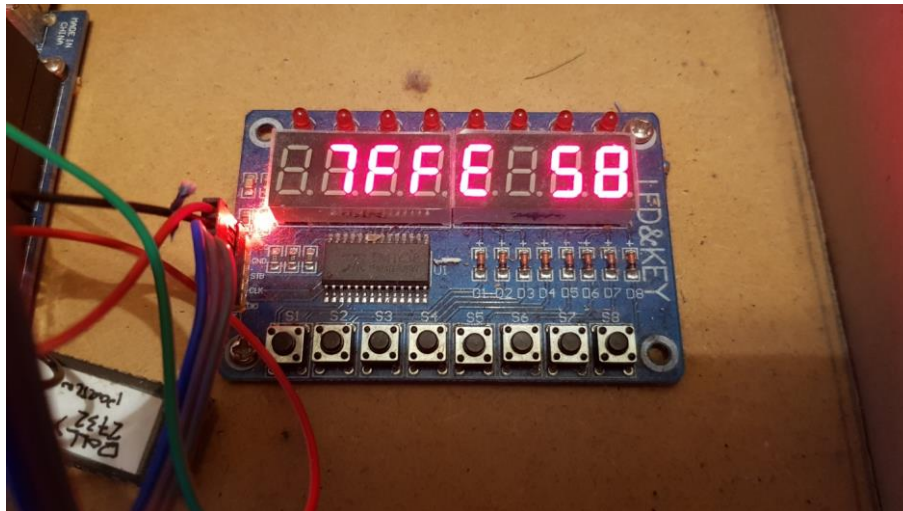
Address bus and data bus are shown after reset. Since A15 is not connected to J5 it floats. Therefore A15 is assumed low.( Thus FFFE shows as 7FFE).

S6 ends test

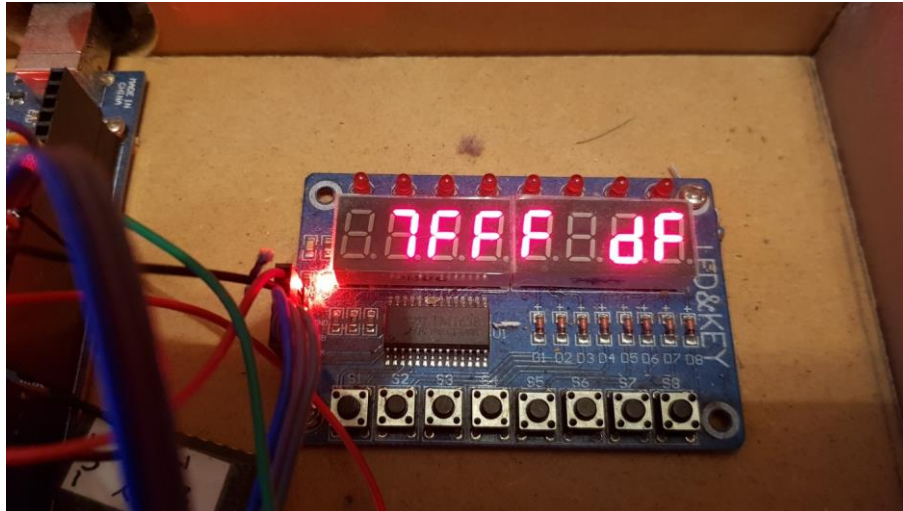
S8 increase to the next address after reset has become high.

S7 show previous address.

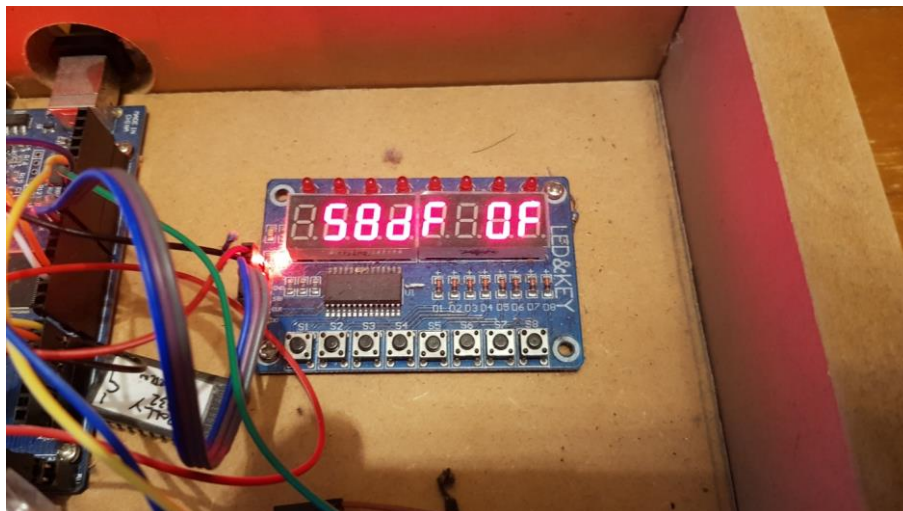
The figure shows: \$7FFE : \$58 first part of reset vector



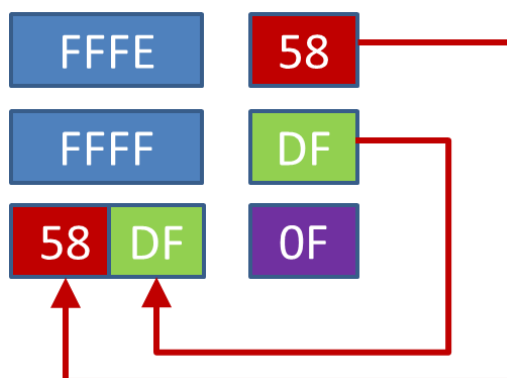
## André's MPU 35 tester



Stepping to the next Address (press S8) gives the first address of program after reset. \$58DF=OF, Opcode 0F (SEI). This means CPU and ROM U6 is working.



The reset vector is stored in FFFE and FFFF and is used to create the first address which is used after reset.





## André's MPU 35 tester

Since A15 is not used and not connected to J5 the address of MPU FFFE will read as 7FFF. After reset the first address should be 7FFE the data in this case 58

The second address should be 7FFF the data in this case DF

The next address should be the concatenation of the 2 data in this case 58DF

The opcode which is then fetched is 0F which is instruction SEI.

Then search for the LDAA 30, STAA \$91 instructions.

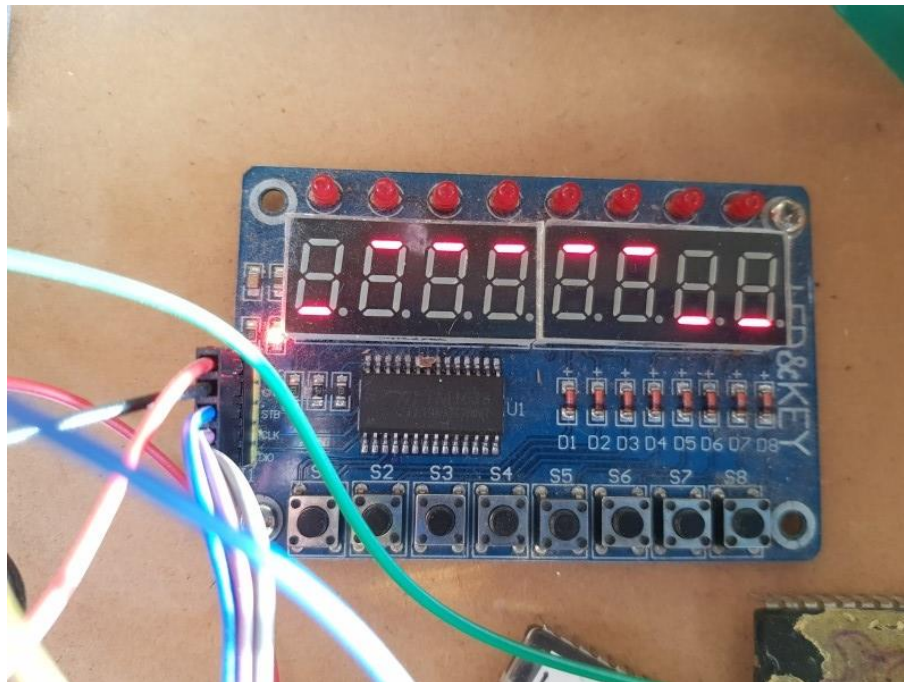
And the address 0091 and data 30 means the pia u11 is being programmed.

If you see this then your MPU is fetching instructions from U6 (if you use 2732 eproms in U2 and U6).

Basically the board should be capable to give a first flicker.

### 5.4 Test 10: Probe readout

Reads probe and display result on led and key. If the probe is attached to right leg of R9 the following should be shown.

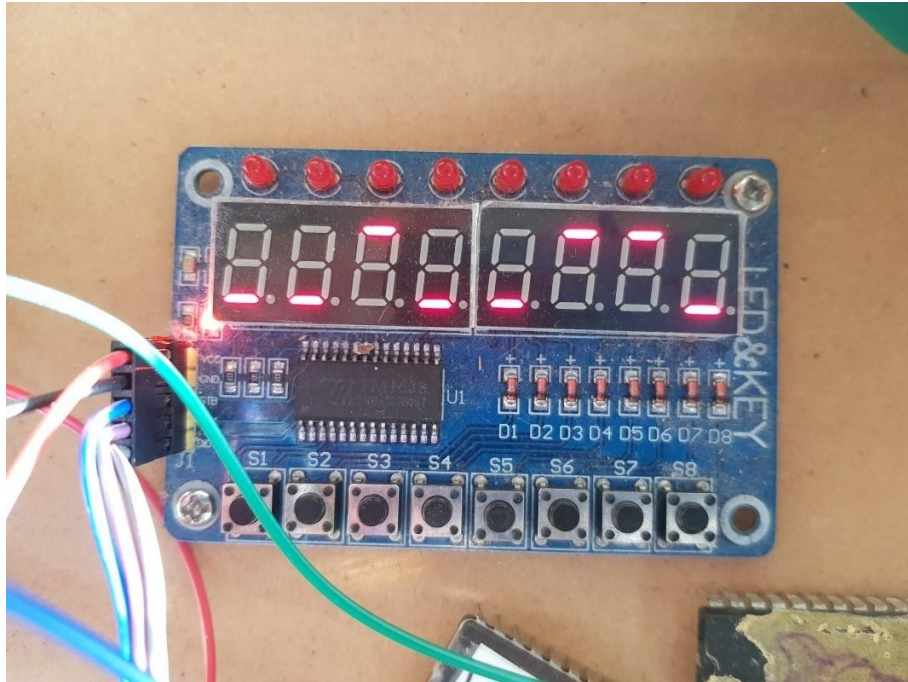


S6 ends test

S7 decrease sample period

S8 increase sample period.

Pressing S8 twice should give something similar to the next picture



## 5.5 Test 11: Address and Datalines

Reads and displays data and address lines.  
S8 will end test.

## 5.6 Test12: Connector J1

All chips can be present, at least the Pia's should be inserted.  
Each pin of connector J1 are "energized" (made 0V) by programming the PIA. Using a led bar connected to the connector it is possible to find a connector problem  
Fi. First pin of J1 is PA7 of U11, in case the led is not lit on the test led bar there is a malfunction.

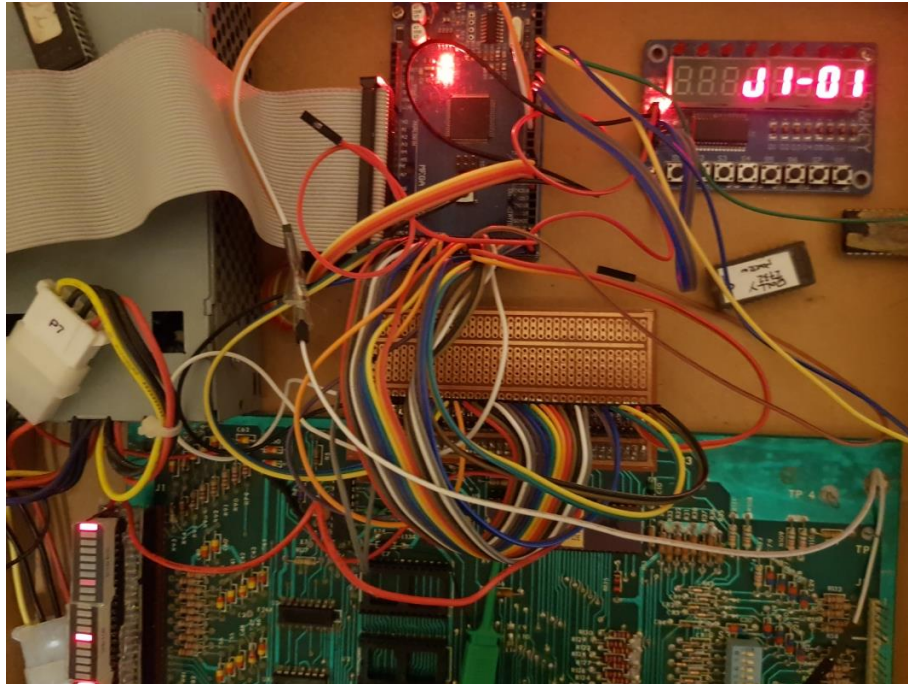
S1 selects test  
S8 next pin  
S7 previous pin.

In general if the Connector tests do not work on a correct MPU board. Reset the MPU board and the MPU tester.

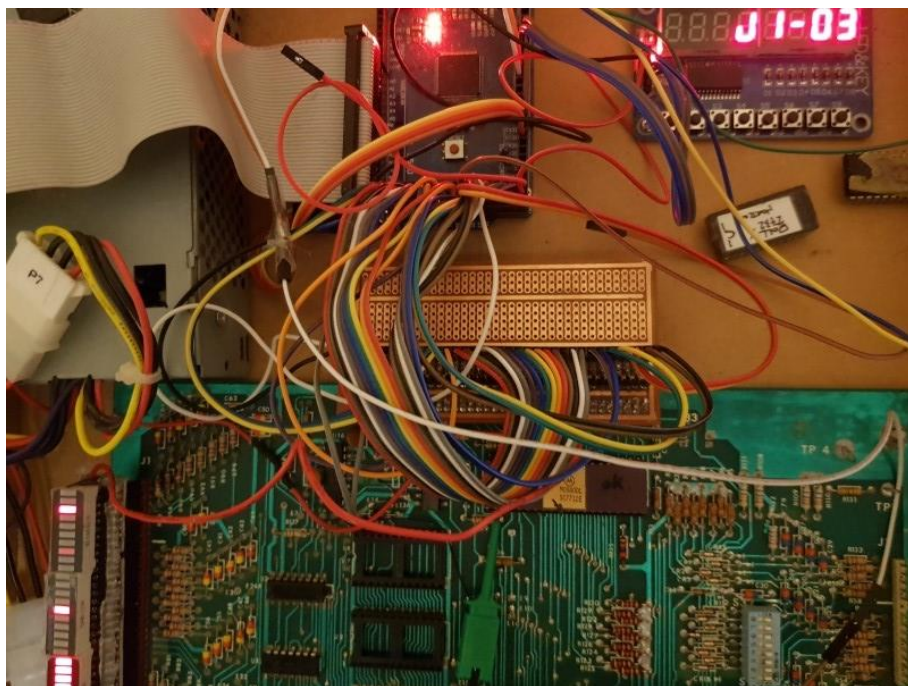
Picture below of a defective board, J1-1 is activated. The First led is on.



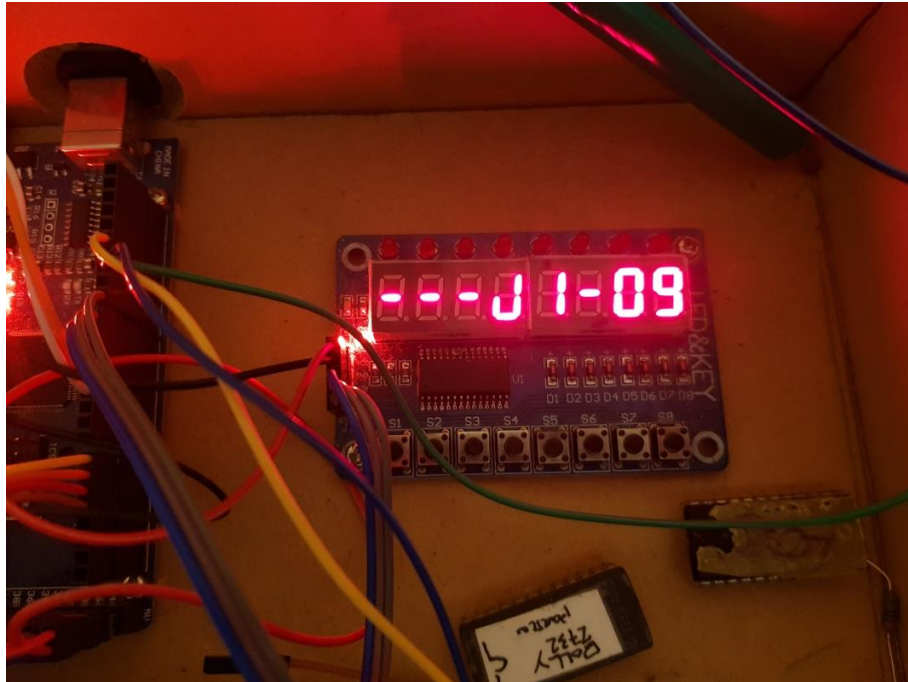
## André's MPU 35 tester



Picture below of a defective board, J1-3 is activated. The third led is on.



If a pin is not connected. The led will display - - - at the start.



## 5.7 Test13: Connector J2

Tests J2.

Important all pia ports are programmed as output for this test. Also J2. The input capabilities are not tested. It can be used to check continuity between pia ports and the connector.

See Connector J1 test for explanation of how to do this test.

## 5.8 Test14: Connector J3

Tests J3.

Important all pia ports are programmed as output for this test. Thus also J3. The input capabilities are not tested. It can be used to check continuity between pia ports and the connector.

See Connector J1 test for explanation of how to do this test.

## 5.9 Test15: Connector J4

See Connector J1 test for explanation of how to do this test.

## 5.10 Test16: Connector J5

**IMPORTANT DO not connect a MPU to the tester for this test**

First step is to test the cable which should go to J5. Do not connect the cable to a Bally MPU J5 yet.

It works the same as the advanced continuity tester. This test should not be used in combination with a MPU. Only to check if the hardware of the MPU-tester and the cable is correct. See also appendix.

### 5.11 Test17: read Phi

A very simple examination of Phi is done.

To check Phi also use the probe test (test 10) and attach it to R8 or R9 or R137 to inspect the clock signal.

### 5.12 Test 18: read Reset

Reset should be a logical 1 (+5V) If the board is powered up.  
Reset 0 means a defect in the reset circuitry.

### 5.13 Test 19: 5101 test

A very slow memory test is executed.  
Each address which is tested is. S6 cancels the test. An error also cancels the test.

### 5.14 Test 20: 6810 test

A very slow memory test is executed.  
The address is shown. S6 cancels the test.  
If an error occurs the test is stopped and "error" is shown.

### 5.15 Test 21: Led on off

Green led is turned off and on. Also  
all pia ports are configured as outputs and are put all on/off.

### 5.16 Test 22: U14 CD4049

Using the probe, the pins of U14 can be tested. The IC needs to be present.  
Using keys S8 to go to next pin. S7 to go back to previous pin.  
To test this IC the green line needs to be connected to TP 3 of the MPU.

### 5.17 Test 23: U15 MC3459L=7437

Using the probe, the pins of U15 can be tested.  
Using keys S8 to go to next pin. S7 to go back to previous pin.  
If a pin is not connected. The led will display - - - before the pin number.

### **5.18 Test 24 through 27 (U16=9602, U17=74L00, U18= 4049, U19= 4011)**

Using the probe, the pins of U16/U17/U18/U19 can be tested.  
Using keys S8 to go to next pin. S7 to go back to previous pin.  
If a pin is not connected. The led will display - - - before the pin number.

### **5.19 Test 28 Display interrupt test.**

Put the probe to pin 3 of U12. Select the test. The frequency is measured of the Display interrupt generator. Correct values are between 160-270 Hz.

## 6. Using a PC to control André's MPU tester

This parts highlights several tests which can be done with the MPU tester. First thing is to test the hardware (cable made to connect the arduino to the MPU J5 connector). This is explained in the appendix. After you made sure the cable is correct connect it to a Bally MPU J5.

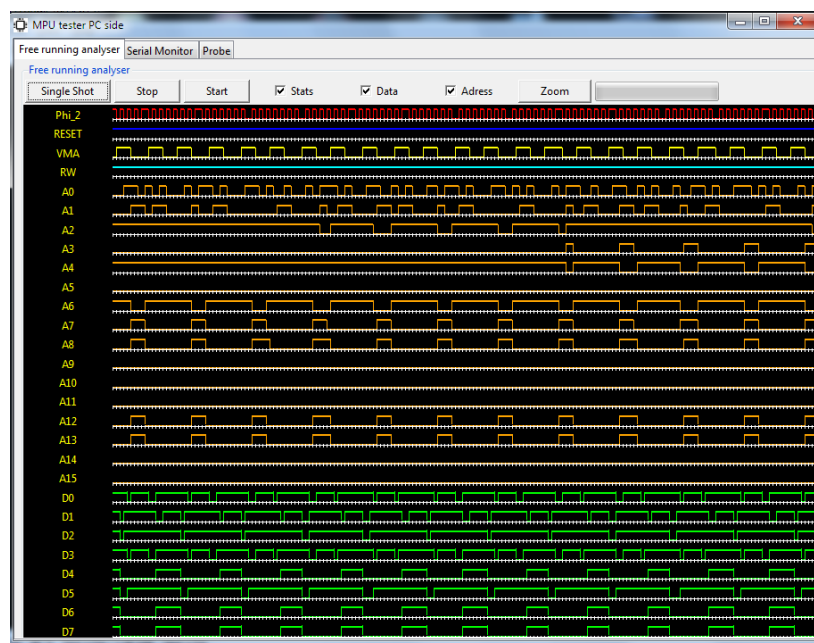
See the section about USB drivers and install them if needed.

### 6.1 PC software to control the mpu tester

Several software packages can be used to control the mpu tester from the pc side. The following paragraphs describes them.

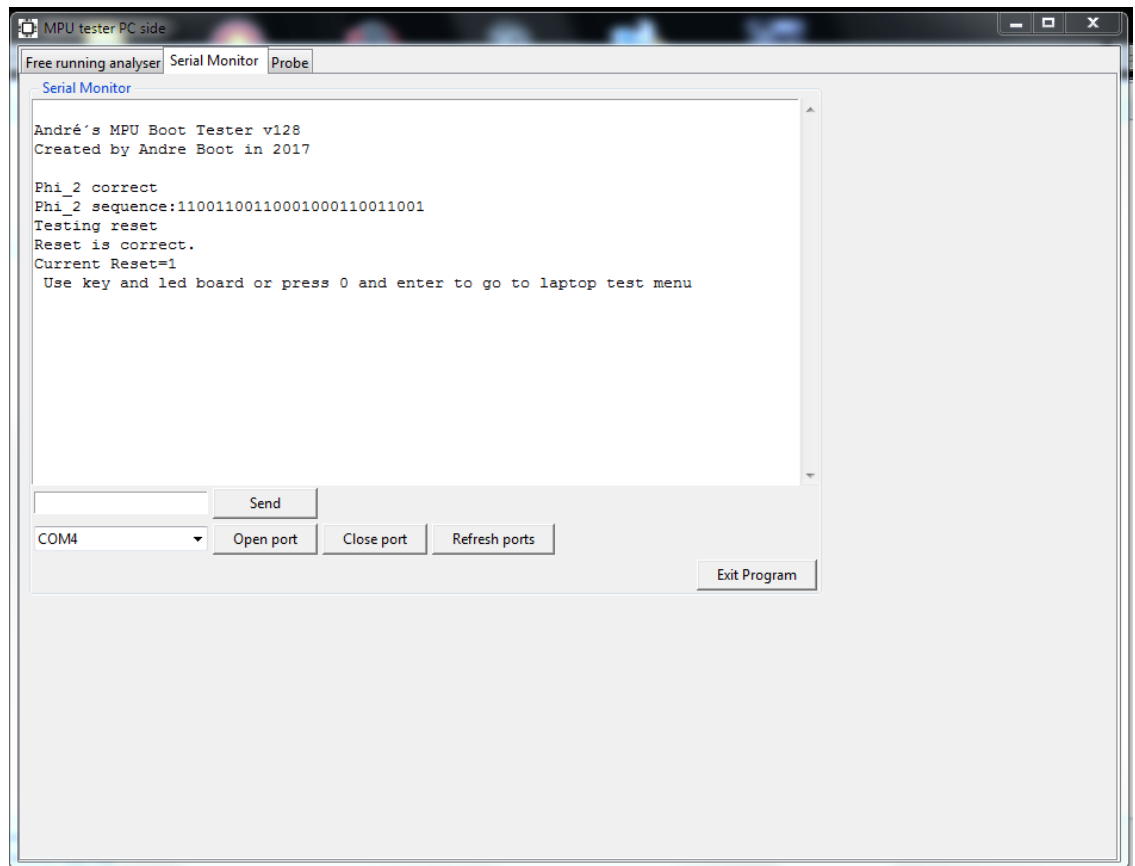
#### 6.1.1 Using andre\_mpu\_gui.exe to control the tester.

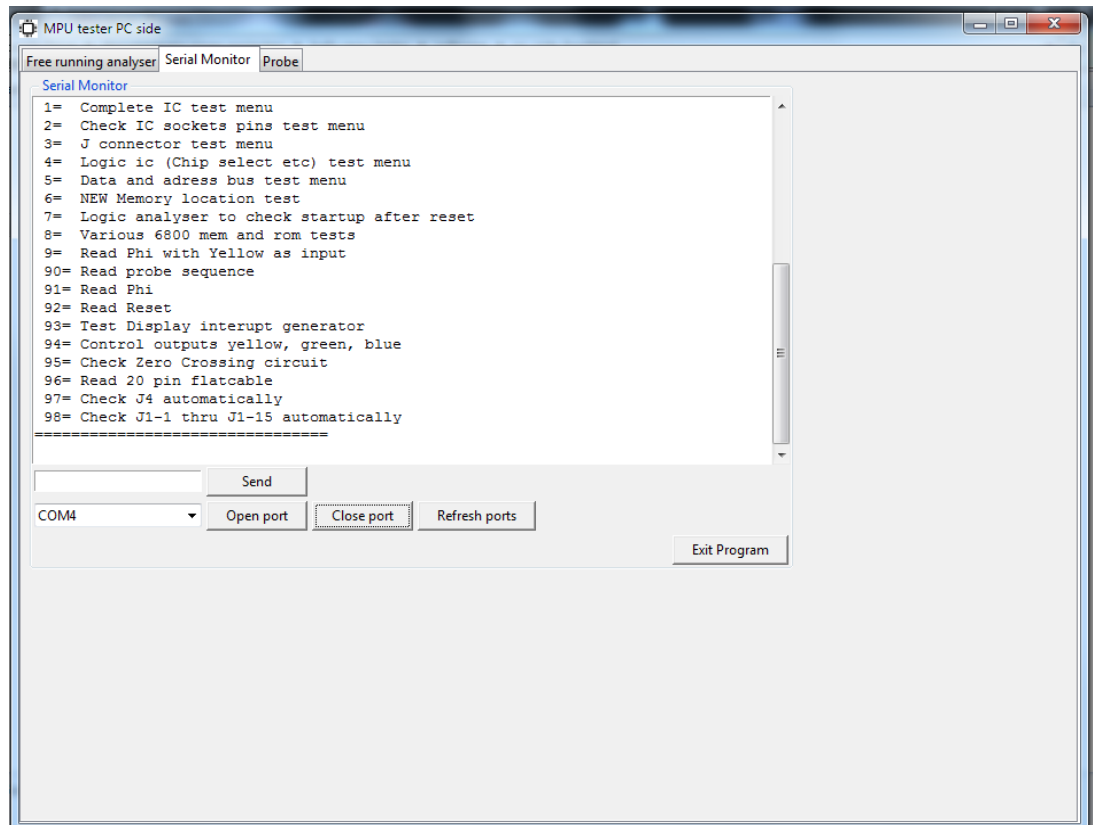
There is also a graphical interface for the PC. andre\_mpu\_gui.exe, if you are lucky it will run. You need both icon2.ico and andre\_mpu\_gui.exe in a directory. It is an executable written in Pyton created on an ancient Windows XP. It is tested on a windows 7. Security wise a big risk. Therefore I am not sure it is without viruses, so please don't use it. If you have a spare laptop without internet capabilities and andre\_mpu\_gui.exe is running, you should see something like the figure below (the first time you have to press single shot two times). Beware it is really slow, after pressing Single Shot to refresh it takes some time. But it will show you all available signals including the right labels which can be helpful.





And the tab serial will show the menu

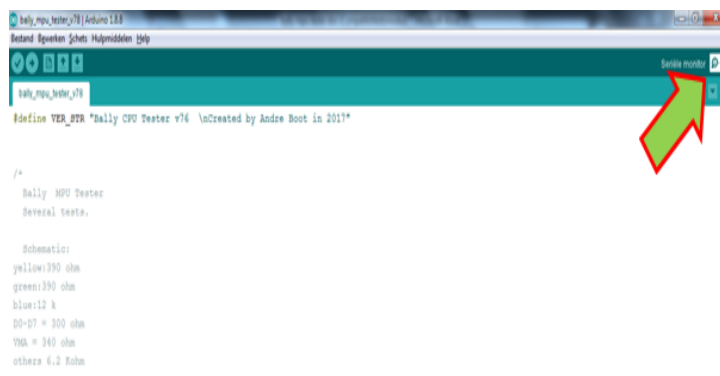




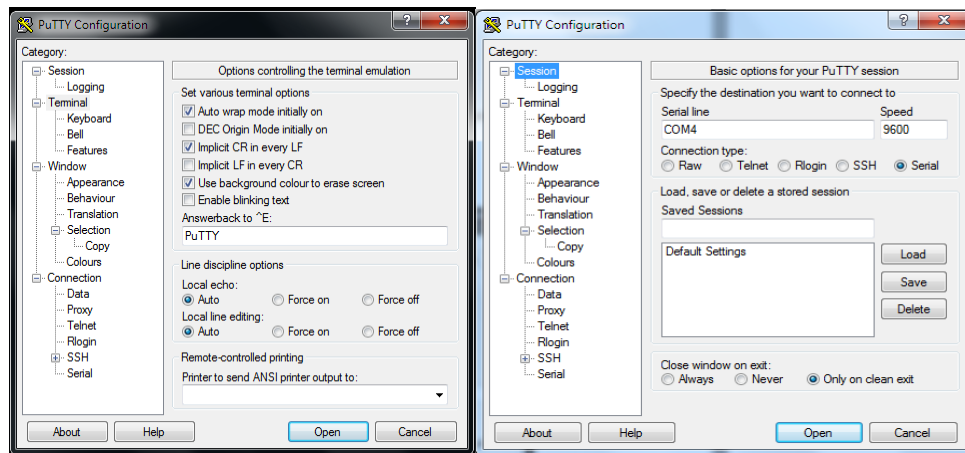
### 6.1.2 Other software to control the tester

Another, safer method, is using other serial software to control the arduino:

Using the arduino IDE serial Reading from the com port



or use putty.exe ((the SSH and Telnet client itself) via <https://www.putty.org/> ). You need to make two modifications to the standard use: Under terminal check box "Implicit CR in every LF", Under session Set it to correct serial parameters, and the correct com port.



And click on Default settings and save them.

## 6.2 Turning on MPU tester

Connect tester to J5 of the MPU. Turn power on the tester and supply power to the MPU from a test bench power supply. After power on and starting putty you will get this or similar message:

```
Bally CPU Tester
Created by Andre Boot in 2017

Phi_2 correct
Phi_2 sequence:00110010001001100110011000
Testing reset
Reset is correct.
Current Reset=1
Use key and led board or press 0 and enter to go to laptop test menu
```

If you see this you should be very happy. Since Phi\_2 is correct and it looks like the reset circuit is ok (at least reset is high).

In this manual only the PC commands are discussed. Thus pressing 0. Will get you in the following or similar test menu.

```
=====
Which Test:
1= Complete IC test menu
2= Check IC sockets pins test menu
3= J connector test menu
4= Logic ic (Chip select etc) test menu
5= Data and adress bus test menu
6= NEW Memory location test
7= Logic analyser to check startup after reset
8= Various 6800 mem and rom tests
9= Read Phi with Yellow as input
90= Read probe sequence
91= Read Phi
92= Read Reset
93= Test Display interrupt generator
94= Control outputs yellow, green, blue
95= Check Zero Crossing circuit
96= Read 20 pin flatcable
97= Check J4 automatically
98= Check J1-1 thru J1-15 automatically
=====
```

To select a menu item press the number. Then a submenu is printed. Which gives the choices to make. It is also possible to run a submenu test directly (for instance selecting 12 will select 6810 test)

It is important to be careful, sometimes it is needed to remove chips. This will be instructed by the test. Test J5 should not be selected at all, since it can actuate datalines which can collide with datalines of fi. The 6800.

### 6.3 Locked MPU (led always on) tips

If a your MPU is locked (led always on).

The following steps could be used to pinpoint the problem:

1. Check if clock is present (Test 91<sup>1</sup>)
2. Check if reset is correct. (Test 92)
3. Check some of the IC's<sup>2</sup>:
  - 6810 (Test 12)
  - 5101 (Test 11)
  - Check for stuck bits(Test 63)
  - LED flash test (Test 192)
  - Do a memory display test of the contents of U6 eprom (see other paragraph)
4. Do an advanced continuity check (often this will point to the error). To be honest this is the reason why this all started, using this test type 2. This is test is explained in following paragraph.
5. Check the logic gate chips U15,U16,U17,U18,U19 (test 290 through 295)

### 6.4 Advanced Continuity Check

Corroded boards soften suffer from lack of continuity. Sometimes in combination with malfunctioning address logic.

**IMPORTANT: Remove all IC's: 6800, 2x6821,5101, 6810, Eproms and proms.**  
**IMPORTANT: Remove all IC's: 6800, 2x6821,5101, 6810, Eproms and proms.**  
**IMPORTANT: Remove all IC's: 6800, 2x6821,5101, 6810, Eproms and proms.**

---

<sup>1</sup> Test 91 and 92 are not needed if you use the graphical pc side software mpu\_tester.exe.

<sup>2</sup> First check if Phi\_2 (clock) is present. Some memory test could hang, because they depend on correct functioning of phi\_2(clock)





If you want to go down in the pin number (fi you are at U8 -18) press S7. To go up again press S8.

Some remarks:

- It will also test pin 19. Which is the signal *not(A9 \*EXT\*VMA\*phi \*not(A12))* Which is created by various logic chips.
- Some pins of fi. U8 9 and 10 are connected to each other. The led will go on if you move from pin 9 to pin 10. Although the led and key will display U8 -9. This is because pin 9 and 10 receive the same signal.
- Some pins cannot be tested (yet) (for instance pia port pins) because they are driven by the pia's.

If you have tested all IC sockets, and the clock is running and and reset is good, you should at least get one flicker/flash if you install the Eprom and MC 6802 and U 11.

## 6.5 Memory display of eprom U6

If it is not clear whether the eprom is correct or reachable. Use memory map for this (test 83)

Type 83, give address FF00. After few lines key in 0 to stop the memory map test. If you see the following:

```

COM5
Verzenden

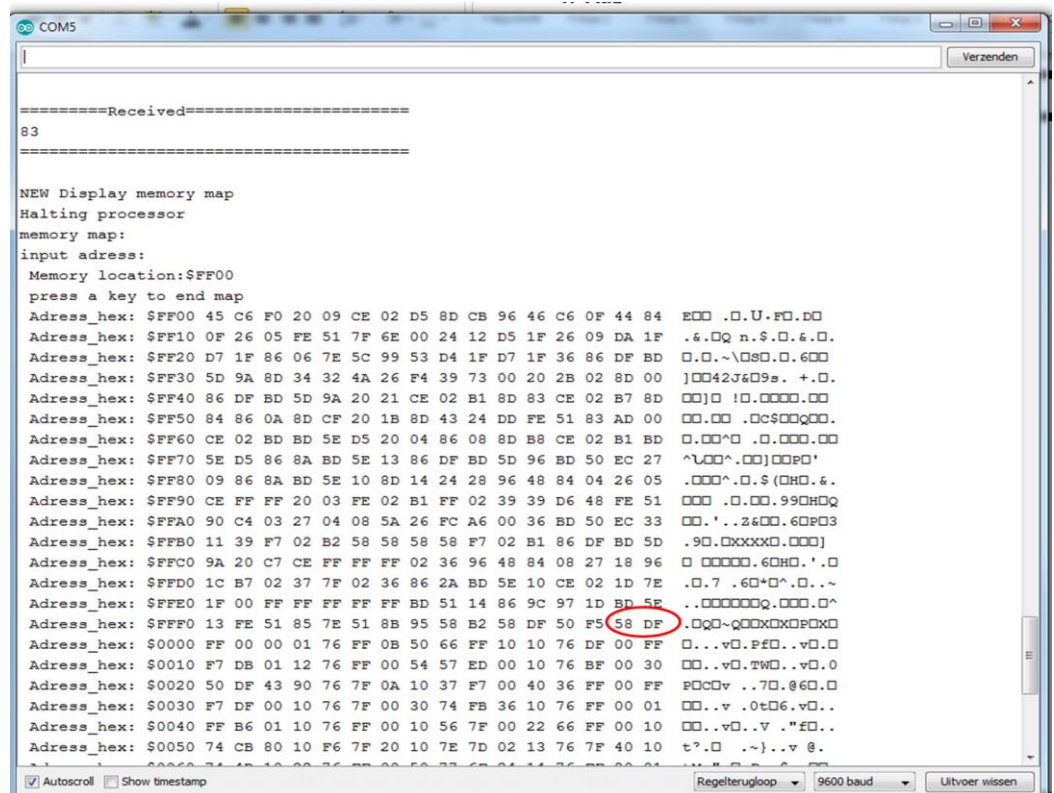
Address_hex: $FF10 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF20 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF30 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF40 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF50 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF60 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF70 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF80 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FF90 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FFA0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FFB0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FFC0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FFD0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FFE0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $FFF0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Address_hex: $0000 FF C9 01 01 76 FF 0B 50 66 C7 10 10 76 DF 00 11
Address_hex: $0010 76 DF 01 1A 76 FF 00 54 47 00 1B 00 1B 00 1B 00
Address_hex: $0020 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00
Address_hex: $0030 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00
Address_hex: $0040 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00
Address_hex: $0050 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00
Address_hex: $0060 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00
Address_hex: $0070 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00 1B 00
Address_hex: $0080 FF FF FF FF FF FF FF FF 1B 00 00 00 1B 00 00 00
Address_hex: $0090 1B B8 1B 00 1B B8 1B 00 1B 00 00 00 1B 00 00 00

Which Test:
1= Complete IC test menu
2= Check IC sockets pins test menu
☒ Autoscroll ☐ Show timestamp
Regelterugloop 9600 baud Uitvoer wissen

```

It not good.

You should see something like this:



```

=====Received=====
83
=====

NEW Display memory map
Halting processor
memory map:
input address:
Memory location:$FF00
press a key to end map
Address_hex: $FF00 45 C6 F0 20 09 CE 02 D5 8D CB 96 46 C6 0F 44 84  E00 .0.U.F0.D0
Address_hex: $FF10 0F 26 05 FE 51 7F 6E 00 24 12 D5 1F 26 09 DA 1F  .6.0Q n.$0.6.0.
Address_hex: $FF20 D7 1F 86 06 7E 5C 99 53 D4 1F D7 1F 36 86 DF BD  0.0.~\080.0.600
Address_hex: $FF30 5D 9A 8D 34 32 4A 26 F4 39 73 00 20 2B 02 8D 00  ]0042J609s.+.0.
Address_hex: $FF40 86 DF BD 5D 9A 20 21 CE 02 B1 8D 83 CE 02 B7 8D  00]0 !0.0000.00
Address_hex: $FF50 84 86 0A 8D CF 20 1B 8D 43 24 DD FE 51 83 AD 00  00.00 .0c$0000.
Address_hex: $FF60 CE 02 BD BD 5E D5 20 04 86 08 8D B8 CE 02 B1 BD  0.00^0 .0.000.00
Address_hex: $FF70 5E D5 86 8A BD 5E 13 86 DF BD 5D 96 BD 50 EC 27  ^000^0.00]0000'
Address_hex: $FF80 09 86 8A BD 5E 10 8D 14 24 28 96 48 84 04 26 05  .000^0.0.$(0H0.6.
Address_hex: $FF90 CE FF FF 20 03 FE 02 B1 FF 02 39 39 D6 48 FE 51  000 .0.00.990H0Q
Address_hex: $FFA0 90 C4 03 27 04 08 5A 26 FC A6 00 36 BD 50 EC 33  00.'..2600.60P03
Address_hex: $FFB0 11 39 F7 02 B2 58 58 58 58 F7 02 B1 86 DF BD 5D  .90.0xxxx0.000]
Address_hex: $FFC0 9A 20 C7 CE FF FF FF 02 36 96 48 84 08 27 18 96  0 00000.60H0.'0
Address_hex: $FFD0 1C B7 02 37 7F 02 36 86 2A BD 5E 10 CE 02 1D 7E  .0.7 .60+0^0.0..^
Address_hex: $FFE0 1F 00 FF FF FF FF FF BD 51 14 86 9C 97 1D B0 5F  ..000000Q.000.0^
Address_hex: $FFF0 13 FE 51 85 7E 51 8B 95 58 B2 58 DF 50 F5 58 DF  .0Q0~Q00X0X0P0X0
Address_hex: $0000 FF 00 00 01 76 FF 0B 50 66 FF 10 10 76 DF 00 FF  0...v0.Pf0..v0.0
Address_hex: $0010 F7 DB 01 12 76 FF 00 54 57 ED 00 10 76 BF 00 30  00..v0.TW0..v0.0
Address_hex: $0020 50 DF 43 90 76 7F 0A 10 37 F7 00 40 36 FF 00 FF  F000v ..70.060.0
Address_hex: $0030 F7 DF 00 10 76 7F 00 30 74 FB 36 10 76 FF 00 01  00..v .0t06.v0..
Address_hex: $0040 FF B6 01 10 76 FF 00 10 56 7F 00 22 66 FF 00 10  00..v0..v ."f0..
Address_hex: $0050 74 CB 80 10 F6 7F 20 10 7E 7D 02 13 76 7F 40 10  t0.0 .~)..v @.
=====
Autoscroll Show timestamp
Regeltemploop 9600 baud Uitvoer wissen

```

Where the Red circle gives you information about the restart vector (In this case there was a Dolly Parton Eprom 2732 inserted at U6) . It gives the reset vector of \$58DF . You can use that to check the startup process with the logic analyser (test 71) see another subsection written about that.

## 6.6 Test 84: Test board without eproms, Load test software into 6810

Are the proms correct, this test can be used to check the MPU without eproms. Remove all EPROMS and roms for this test. So if you got faulty eproms, but want to know whether the 6800 and 6810 and pia U11 is working this is a useful test: Remove eproms. Insert 6800, 6821 U11, and 6810 . The ic 5101 may also be present.

This test loads a "blink" and test software into the RAM IC 6810.

Select 84.

After the test, press reset on the MPU board.

If the 5101 is correct It should display

First led (J4-1) on

Second led (J4-1) on'

After that all leds and MPU green led start a on off cycle.

If the 5101 is not should display

All leds and MPU green led start a on off cycle.

This test seems sometimes to unable to load to the chip.  
But when it does it shows you that 6800,6810,6821 U11 and 5101 are correct.

For the interested. If no eprom is inserted. Address FFFF and FFFE are both FF. The resetvector becomes FFFF. The program counter starts at FFFF. It will read instruction FF 00 7E (STX \$007E) and the next instruction is executed at \$0003 which holds the rest of test program.

For the 6800 programmers: \$80 bytes of code minus 4 bytes stack was challenging.

## 6.7 Logic analyzer function checking startup

Only a few test are possible with roms/and mc6800 chips present.  
So if all chips are present, select the logic analyzer function.

Select 7.

=====Menu Logic analyser =====

71= state Logic analyser to check startup after reset  
72= state Logic analyser at arm adress  
73= time Logic analyser to check startup after reset  
74= time Logic analyser adress lines only  
75= time Logic analyser data lines only  
76= time Logic analyser stat lines only  
77= time Logic analyser at arm adress

=====

Only the option 71 is described:

Make sure the reset is low by switching the 12 V supply to the board off.  
Press a key, if you see the text "waiting", supply the 12V to the board. After that it should look like this:

## André's MPU 35 tester

Logic analyser (state) to check startup after reset

First press a key and then release reset

Press a key to end analyser

Press a key to start analyser

PRESS A KEY and ENTER

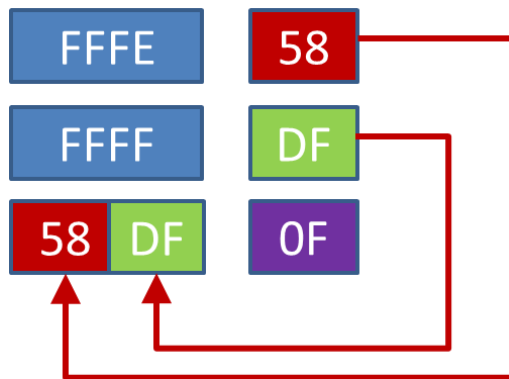
Waiting

Adress :Data : Opcode length: Opcode: Operand

```

7FFE : 58 : [1] ASLB : Reset vector High Byte
7FFF : DF : [2] STX : 0F Reset vector Low Byte
58DF : 0F : [1] SEI : Start of Program
58E0 : 86 : [2] LDAA : 86
58E0 : 86 : [2] LDAA : 30
58E1 : 30 : [1] TSX :
58E2 : 97 : [2] STAA : 91
58E3 : 91 : [2] CMPA : 30
0091 : 30 : [1] TSX :
58E4 : CE : [3] LDX : 55F0
58E5 : 55 : [0] ??? :
58E6 : F0 : [3] SUBB : 0926
58E7 : 09 : [1] DEX :
58E8 : 26 : [2] BNE : 26
58E8 : 26 : [2] BNE : FD
58E9 : FD : [0] ??? :
58E7 : 09 : [1] DEX :
58E8 : 26 : [2] BNE : 26
58E8 : 26 : [2] BNE : FD

```



Since A15 is not used and not connected to J5 the address of MPU FFFE will read as 7FFF. After reset the first address should be 7FFE the data in this case 58

The second address should be 7FFF the data in this case DF

The next address should be the concatenation of the 2 data in this case 58DF  
The opcode which is fetched is 0F which is instruction SEI.

Then search for the LDAA 30, STAA \$91 instructions.

And the address 0091 and data 30 means the pia u11 is being programmed.

If you see this, your MPU is fetching instructions from U6 (if you use 2732 eproms in U2 and U6).

Basically the board should be capable to give a first flash/flicker.

## 6.8 IC tests

Menu 1 is for "complete IC test"

Select 11 and 12 to check the memory chips.

And play with the others, but they are not completed thus not described further.

## 6.9 Test 83: Display memory

This can be used to view the contents of the EPROM and other memory locations.

If address \$FFF0 is can be used to read the reset vector.

# 7. Appendix

## 7.1 Testing the J5-cable

First step is to test the cable which should go to J5. Do not connect the cable to a Bally MPU J5 yet.

## André's MPU 35 tester

Turn on the power to the Arduino mega, connect it to the USB, launch putty.

Key in :0

Key in :3

Key in :31

This selects test 31.

Then use the probe to verify the connector is correct.

It works the same as the advanced continuity tester. Which is explained in another paragraph.

Led and key should show you the following J5 -1.

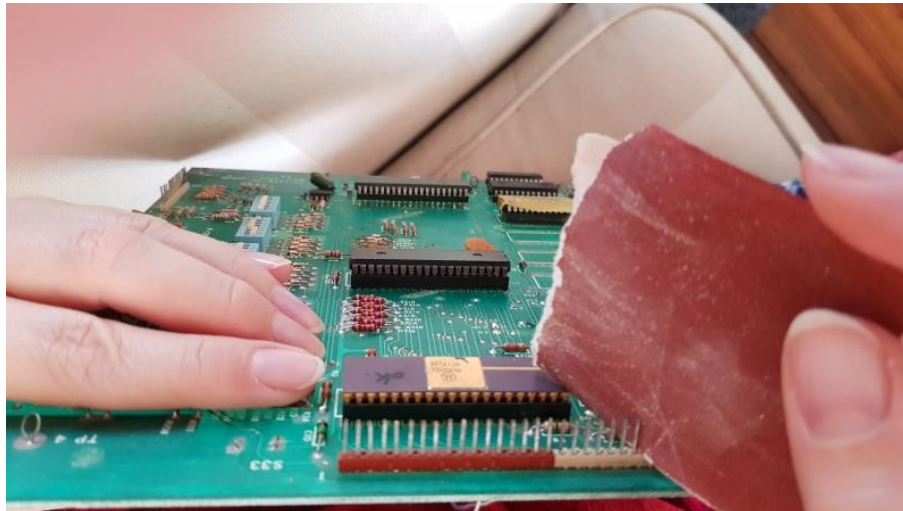
Take the probe (in my case the white wire). Connect it to pin 1. The 2nd led of the "Led and key" will be lite up, if there is continuity to this pin.

Put the probe at pin 2. Normally the led would go off.

Press S8. The led and key will display the next pint its actuating and..the led should go on. Repeat until last pin.

Press S8 for the next pin until last pin is reached. If you press S8 again you will return to the menu on the PC.

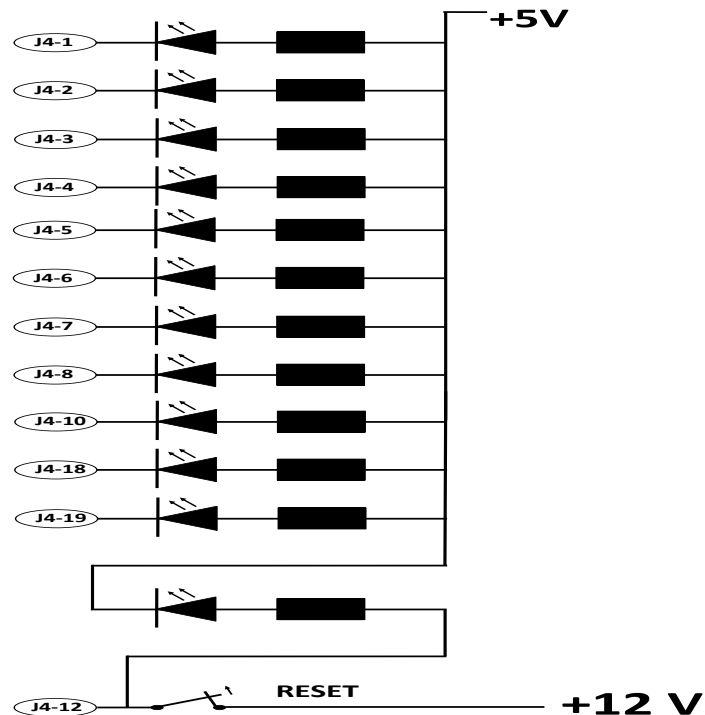
It is important to sand the connector J5 of the bally board, also between the teeth if you are using cheap Chinese connectors, which connect to the "left and right side" of the pins.





## 7.2 J4 connector interface

The J4 connector needs to be supplied with 5 and a switchable 12 V.



Using some tests it is possible to test the output of PB0 of U11 of the Pia. Therefore some leds and resistors are added. Resistors 1 Kohm.

## 7.3 avr\_dude info.

Programming the arduino mega :

avr dude command the arduino ide used was:  
 C:\Program Files (x86)\Arduino\hardware\tools\avr\bin/avrdude -CC:\Program Files (x86)\Arduino\hardware\tools\avr/etc/avrdude.conf -v -patmega2560 -cwiring -PCOM4 -b115200 -D -Uflash:w:C:\Users\Andre\AppData\Local\Temp\arduino\_build\_147573\bally\_mpu\_tester\_v78.ino.hex:i

## 7.4 Arduino code snippet declaring I/O

cryptic snip off the code where the pins are listed

```
int read_1 = 2;
int anti_line = 3;
```

```
//led and key

const int strobe = 6; //6 PH3
const int clock = 7; //7 PH4
const int data = 8; // going to be 8 PH5

int Out_yellow = 12; // going to 12 PB6 /oc1b
int Out_green = 11; // going to 11 PB5
int Out_blue = 10; // going to 10 PB4

int A_14    =31;
int U11_pin18 =39;
int Ext_mem  =41;
int Phi_2    =53; // PB0
int VMA      =52; // PB1
int nRes     =50; // PB3
int HLT      =40;
int r_w      =51;
int A_0      =49;
int A_1      =48;
int A_2      =47;
int A_3      =46;
int A_4      =45;
int A_5      =44;
int A_6      =43;
int A_7      =42;

int A_8      =37;
int A_9      =36;
int A_10     =35;
int A_11     =34;
int A_12     =33;
int A_13     =32;

int D_0      =22;
int D_1      =23;
int D_2      =24;
int D_3      =25;
int D_4      =26;
int D_5      =27;
int D_6      =28;
int D_7      =29;
```