Brugervejledning til

Dynamic RAM Test Program

23-11-83 Supermax

Af: Lars Bo Pedersen dansk data elektronik a/s

For fejlfri funktion forudsættes det, at man har et fejlfrit CPU-kort og et fejlfrit RAM-kort til rådighed. Det RAM-kort,der ønskes testet forsynes med en PAL C182 og adresseres dermed fra adresse 200000 til 2FFFFF. De to EPROM´er indeholdende UNIBUG og testprogrammet sættes i CPU-kortet, en terminal tilsluttes serviceporten og testen kan begynde.

Beskrivelse af testprogrammet.

Testprogrammet består af to afsnit:

A. Hurtig test.
B. Udvidet test.

Når en af disse to test startes svarer programmet med følgende udskrift:

```
***** DYNAMIC RAM TEST PROGRAM *****
*****   23-11-83   SUPERMAX     *****
```

A. Hurtig test.

Denne test skriver først Low Word af adressen, skiftet en bit til højre i den tilsvarende adresse. Derefter kontrollæses og i tilfælde af fejl udskrives disse på terminalen. Derefter skrives High Word af adressen i den tilsvarende adresse og der kontrollæses. Denne test vil i langt de fleste tilfælde finde fejlen på RAM-kortet. Testen svarer med følgende udskrift:

```
HIGH SPEED TEST
START ADDRESS: XXXXXXXX          END ADDRESS: YYYYYYYY
    ***** TEST COMPLETED *****
```

B. Udvidet test.

Denne test er mere effektiv end den foregående, den tager til gængæld også noget længere tid. Testen består af tre delprogrammer:

1. Bittest.

Et enkelt bit sættes og skrives sandt og inverteret i hele lageret. F.eks. første gennemløb skrives 0001, derefter kontrollæses, andet gennemløb skrives FFFE, derefter kontrollæses, tredie gennemløb skrives 0002, osv. ialt 32 gennemløb. Denne test svarer med følgende udskrift:

```
BIT TEST
START ADDRESS: XXXXXXXX          END ADDRESS: YYYYYYYY
++++++++++++++++++++++++++++++++ TEST FINISHED
```

2. Adressebittest.

Hvert enkelt adressebit testes ved at skrive følgende bitmønstre i lageret. Testen gennemløbes 4 gange, hver med forskelligt bitmønster.

	1.	2.	3.	4.
Adressebit = 0 skrives	0000	FFFF	0000	8001
Adressebit = 1 skrives	FFFF	0000	8001	0000

Testen gennemføres for adressebit 1-19. Test 3 og 4 tester fejlkorrektionsbittene. Testen svarer med følgende udskrift:

```
ADDRESS BIT TEST
START ADDRESS: XXXXXXXX          END ADDRESS: YYYYYYYY
TESTING ADDRESS BIT £ ZZ
TESTING ADDRESS BIT £ ZZ
TESTING ADDRESS BIT £ ZZ         ZZ er adressebitnummer på hex form.
TESTING ADDRESS BIT £ ZZ         ZZ tælles fra 01 til 13.
  TEST FINISHED
```

3. Galloperende nuller og ettaller.

Testen gennemløbes 4 gange. Først initialiseres hele lageret med en bestemt baggrundsværdi. Derefter skrives en galloperende værdi i en bestemt adresse.

	1.	2.	3.	4.
Baggrundsværdi	0000	FFFF	0000	8001
Galloperende værdi	FFFF	0000	8001	0000

Herefter testes alle naboceller til den celle, hvor den galloperende værdi er skrevet. Nabocelle vil sige at adressen kun afviger på en bit. Nabocellerne skal stadig indeholde den initialiserede værdi. Dette gentages for alle adresser. Testen svarer med følgende udskrift:

```
GALLOPPING ONES AND ZEROES
START ADDRESS: XXXXXXXX          END ADDRESS: YYYYYYYY
GALLOPPING VALUE: FFFF
GALLOPPING VALUE: 0000
GALLOPPING VALUE: 8001
GALLOPPING VALUE: 0000
  TEST FINISHED
    ***** TEST COMPLETED *****
```

Kommandoer for afvikling af programmet.

Følgende sekvens udføres:

*GO 30A6	MMU værdier initialiseres
*GO 3000	MMU værdier skrives
*A0 200000	Startadresse i A0
*A1 2FFFFF	Slutadresse i A1

Derefter startes en af de to test A eller B.

*GO 34C8	Start hurtig test
*GO 3266	Start udvidet test

Hvis en test afbrydes skal den startes igen med en af de to ovennævnte ordrer.

Fejludskrifter.

A. Hurtig test.
Hvis der under hurtigtesten bliver fundet fejl, svarer programmet med følgende udskrift:

ERROR IN ADDRESS: XXXXXXXX WRITE DATA: YYYYYYYY READ DATA: ZZZZZZZZ

XXXXXXXX er den adresse, hvor fejlen er fundet.
YYYYYYYY er de data, der er skrevet på adressen.
ZZZZZZZZ er de data, der er læst på adressen.

Testen afbrydes hvis der findes 10 fejl. Hvis dette er tilfældet svarer programmet med:

10 ERRORS. TEST CANCELED*

Da testen kører med fejlkorrektion enabled, vil enkelt fejl blive rettet og dobbeltfejl giver anledning til bus trap error. Hvis det sidste er tilfældet, køres den udvidede test. Hvis enkelt fejl er blevet rettet, svarer programmet med:

ERROR HAS BEEN CORRECTED
ROW NUMBER: X SYNDROME BITS: 00YY

X er rækkenummer hvor fejlen er rettet.
YY fortæller hvilken kreds i rækken, hvor fejlen er rettet.

Bag i denne vejledning findes en tegning af RAM-kortet, der viser hvordan rækkenummer og syndromebits hænger sammen med placering på kortet. Hvis man får denne fejl skiftes den udpegede kreds og testen køres igen, da programmet kun fortæller om den sidst rettede fejl.

B. Udvidet test.

1. Bit test.
Denne test kører med fejlkorrektion disabled, og har følgende fejl-udskrifter med samme betydning som ovenfor.

ERROR IN ADDRESS: XXXXXXXX WRITE DATA: YYYYYYYY READ DATA: ZZZZZZZZ

Denne test afbrydes også med følgende udskrift, hvis der findes 10 fejl.

10 ERRORS. TEST CANCELED*

2. Adressebit test.
Denne test kører med fejlkorrektion enabled og fejludskrifterne har samme betydning som under hurtigtesten.

ERROR IN ADDRESS: XXXXXXXX WRITE DATA: YYYYYYYY READ DATA: ZZZZZZZZ

ERROR HAS BEEN CORRECTED
ROW NUMBER: X SYNDROME BITS: 00YY

Denne test afbrydes også med følgende udskrift, hvis der findes 10 fejl.

10 ERRORS. TEST CANCELED*

3. Galloperende nuller og ettaller.
Denne test kører med fejlkorrektion enabled og fejludskrifterne har samme betydning som under hurtigtesten.

ERROR IN ADDRESS: XXXXXXXX WRITE DATA: YYYYYYYY READ DATA: ZZZZZZZZ

ERROR HAS BEEN CORRECTED
ROW NUMBER: X SYNDROME BITS: 00YY

Udover disse udskrifter har denne test følgende udskrifter:

ERROR WHEN READING BACKGROUND VALUE
ERROR WHEN READING GALLOPPING VALUE

Disse udskrifter oplyser om de omstændigheder, hvorunder den efterfølgende fejl blev fundet.

Stop programafviklingen.

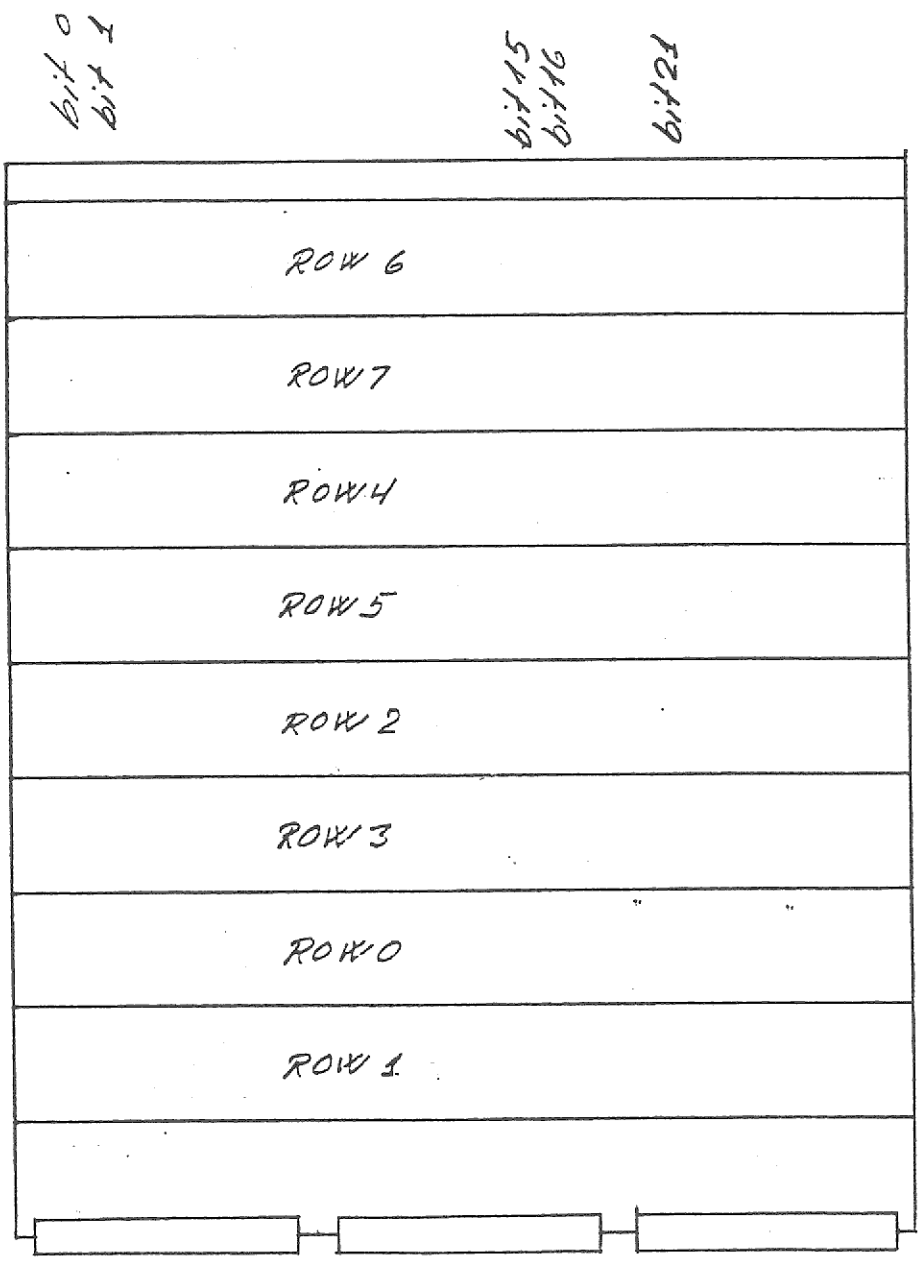
Hvis man f.eks. på grund af mange fejl ønsker at stoppe programafviklingen, trykkes på "ESC", og programmet standser med følgende udskrift:

** TEST CANCELED BY ESC

Der kan gå forholdsvis lang tid, fra man trykker på "ESC" til testen standser, idet den igangværende deltest færdiggøres.

NB. Testen kan IKKE fortsættes fra det sted den blev afbrudt.

Databit	Syndromebit
0	74
1	72
2	71
3	6C
4	6A
5	69
6	65
7	63
8	5C
9	5A
10	56
11	55
12	53
13	4E
14	4D
15	4B
16	7E
17	7D
18	7B
19	77
20	6F
21	5F


bit 0
bit 1
bit 15
bit 16
bit 21
ROW 6
ROW 7
ROW 4
ROW 5
ROW 2
ROW 3
ROW 0
ROW 1

****

Diagnostic program for

CPU Module 0100

RAM Module 0200

Version 84-05-01

****

840501 lars bo pedersen

dansk data elektronik a/s

****
******* Diagnostic programs for SUPERMAX card family ********
****

General information

All intelligent modules in a SUPERMAX are supplied with a service port. The service port is used by the diagnostic programs, and is an RS 232-C interface modified to support parallel connection between all modules.

The test programs run separately in each unit. Consequently it is possible to start programs in all units controlled by one terminal.

In the version of the program available 84-05-01 it is necessary to replace normal boot PROM´s with PROM´s containing test programs. In later versions the PROM will contain both the boot loader program and the test program, and it will be possible to test the communication between the different units.

To ensure that only one service port sends messages to the terminal, only one unit is selected at a time. All units are deselected upon start of the test programs. To select a specific unit, type the following sequence on the terminal:

```
ESC
=
<unit number in hex code> (0-F)
<CR>
```

If the unit is ready to be selected it will respond:

```
**XXXX Test Program **
** Version XX-XX-XX **
** Unit Number  X    **
```

If a unit does not respond to the select sequence the cause of the trouble might be one of the following:

1. The unit is not installed in the card cage.
2. The unit is not connected to the serviceport.
3. The test program is not installed in the PROM.
4. The unit has failed in a way that makes it unable to run the test program.

If you have any units connected in parallel be sure that you have a service port pull down resistor mounted in one and only one unit.

Diagnostic program for CPU Module 0100 and RAM Module 0200

The purpose of this program is to evaluate the basic functions of the CPU module and the RAM module in the Supermax card family.

The following functions can be tested with this program:

The dynamic RAM connected to the CPU.
The Memory Management Unit (MMU).
All interrupts.
All R/W registers.
The Error Detection and Correction unit.

You have to make some changes to the straps and the PCB before the test program is able to run on the CPU. The interrupt strap Z5 mounted in position B 10,5 must contain a connection from pin 6 to pin 10. Pin 10 on the 74LS05 called X6 must be disconnected from the PCB. This component is mounted in position B 10,4.
When you are sure that those changes are made you should be able to run the testprogram. You have to substitute the boot proms with two containing the CPU test program. Connect a terminal to the serviceport. Press reset button and wait for the display on the front panel to turn off.

Selftest after reset.

After pressing reset a small selftest will be performed. By looking at the display on the front panel it is possible to follow program execution especially when a fault occurs. If there are no errors during selftest you will see a ´hd´ in the display and as soon as the PROM is copied to the RAM you will see a ´oo´ in the display. The CPU is ready to be selected when the display is turned off. If an error occurs the display will change depending upon the error.

E1 in display indicates error in MMU memory test.
E2 in display indicates error in Address Space Register.
E3 in display indicates error in Interrupt Mask Register.
E4 in display indicates error in main memory 0-4000.
bE in display indicates a bus error.

If one of the error messages has been displayed you should try to type the select sequence. It is possible that you might get some more information about the error.

How to operate the test program.

Now you can type in the select sequence for the CPU you want to test, and it should respond with the following message on the screen:

```
** CPU  Test Program**
** Version 84-05-01 **
**  Unit Number  X  **
**  Y.YY Mb memory  **
```

This message indicates that the selftest was successfully completed and that you can proceed with the more sophisticated parts of the test.

Following this message you will get the menu from which you can select several tests.

A - Test RAM. High Speed

This routine performs a fast test of the dynamic RAM connected to the CPU module. It takes the low order word of the current address and shifts it one bit to the right and writes that value in memory. Then the memory content is read and compared to the value written and errors are indicated. All of this is repeated using the high order word of the current address.

B - Test RAM Bit Test

During this test every single bit in the main memory will be written with both true and inverted values. This is done using long word operations but on a word basis which means that during pass one the value written is 00010001, during pass 2 it is FFFEFFFE, during pass 3 it is 00020002, during pass 4 it is FFFDFFFD etc. All in all 32 passes. A plus (+) is written on the terminal for each pass.

C - Test RAM. Address Bit Test

During this test every single address bit is tested. This is done by writing one value in the current address if the selected address bit is zero and another value if the address bit is one. The test is performed for address bit in the range 1-19 and the test runs four times, each time with two new write values.

Address bit	Pass 1	Pass 2	Pass 3	Pass 4
0	0000	FFFF	0000	8001
1	FFFF	0000	8001	0000

The reason for the value 8001 during pass 3 and 4 is to be sure that the bits concerning the Error Detection and Correction unit also will be tested.

You should notice that the program writes the current address bit number on the terminal and that this number is on hexadecimal form, which means that the count values are 01-09, 0A-0F and 10-13.

D - Test RAM. Gallopping Ones And Zeroes

This test is also performed 4 times. First of all the memory space is initialized with a default value. Then a gallopping value is written in a certain address. Then all addresses which are neighbours to that first address are tested. They must still contain the default value. A neighbour is an address which only differs in one bit from the original address. When all neighbours are tested the default value is written back in the original address. This is repeated for all addresses.

	Pass 1	Pass 2	Pass 3	Pass 4
Default value	0000	FFFF	0000	8001
Gallopping value	FFFF	0000	8001	0000

E - Test RAM. Byte Write

Because of the construction of the RAM module it is much more difficult to write a byte in memory than it is to write a word. 16 bits are used to contain data and 6 bits contain the ECC code belonging to that data. If one byte (8 bits) is to be written you have to read the whole 16 bit word into a register, then swap the new byte in its place and write the whole word back in memory. During this test memory is initialized by writing a byte counter. Then the value 55 is written in all even memory locations and the whole memory is tested for correct content. The memory is then initialized again and the value 55 is written in all odd bytes. The whole memory is again tested for correct content. All write functions use byte write operations.

F - Test RAM. Read Modify Write

The MC68000 microprocessor has a feature which makes it possible in one operation to read a byte in memory, test the byte, set one bit in the byte and to write it back again. This is called read modify write and this test will initialize memory with a default value and then perform the Test And Set (TAS) on all memory locations and then check for correct memory content.

J - Test MMU Memory

The MMU memory consists of eight 1024x4 static RAMs. This memory can be tested using the same routines as those for the dynamic RAM, except the two using byte write operations. When you select this test the above mentioned tests A, B, C and D will be performed on the MMU memory.

K - Test MMU Adder

The 16 bit adder is build from four 4 bit adders, which can be tested separately. This routine tests each 4 bit section of the adder in the MMU. The main memory is initialized in such a way that each double word in memory contains its own physical address. The memory is accessed using the relocating capabilities of the MMU. The physical address is calculated by adding the offset stored in the MMU to the logical address used when accessing the main memory. The calculated physical address is compared to the real physical address as read in the main memory.

L - Test MMU ASN con SEG

This routine tests that the MMU is addressed correctly by Address Space Number and Segment Number. The content of the main memory is initialized with physical addresses. The MMU memory is initialized with increasing offsets. For all Address Space Numbers and Segment Numbers the main memory is accessed using the relocating capabilities of the MMU. The calculated physical address is compared to the real physical address as read in the main memory.

M - Test MMU Comparator

This routine tests the 12 bit comparator used in the memory protection capabilities of the MMU. It accesses main memory using different Block Numbers (BN) and different Sizes in the MMU while the checking facilities of the MMU are enabled. Whenever BN > Size the memory access is illegal and a bus error should

occur and if BN <= Size the memory access is legal and no bus error should occur.

I - Test Interrupts

This routine tests the interrupt circuit by setting all six levels of external interrupts and checking that they arrive at the CPU. The timer interrupt is also tested and you can calculate the time between timer interupts by measuring the time between two writings of "Timer interrupt" on the terminal and then divide that time by 100. Normally you should end up with 40 ms.

R - Test Registers

There are two read/write registers on the CPU module, the Address Space Register and the Interrupt Mask Register, and they are tested for correct read and write operation.

O - Repeat test(s)

All of the above mentioned tests will run one time through when selected. If when selecting test(s) this one (O) is also selected the test(s) will repeat itself (themselves) until otherwise canceled.

S - Test All

When you select this command, all of the above mentioned test will be executed in the order written. You can also add the "Repeat test" option to this one.

Q - Reset CPU

This command will start the program again in the same manner as if the reset button had been pressed, but it only affects the unit currently selected. After doing this the unit will end up being not selected, which means you have to type the select sequence one more time if you wish to continue testing that unit.

Select test(s):

The program will ask you to select test(s) and the only thing to be aware of is to separate the selected tests by commas.

Start Address/End Address

If you have selected A, B, C, D, E or F you will be asked to type in start address and end address for the memory test. It has to be on hexadecimal form. You will not be allowed to type in illegal addresses.

The DEL or RUB key will cancel all previous pressed keys.

How to control the program

You are able to control program execution by using some special control characters as mentioned below. They do not affect programs running on a unit which is not selected.

Control C

This is the one to use to cancel a running test. Program execution stops and you will return to menu after pressing <CR>.

Control E

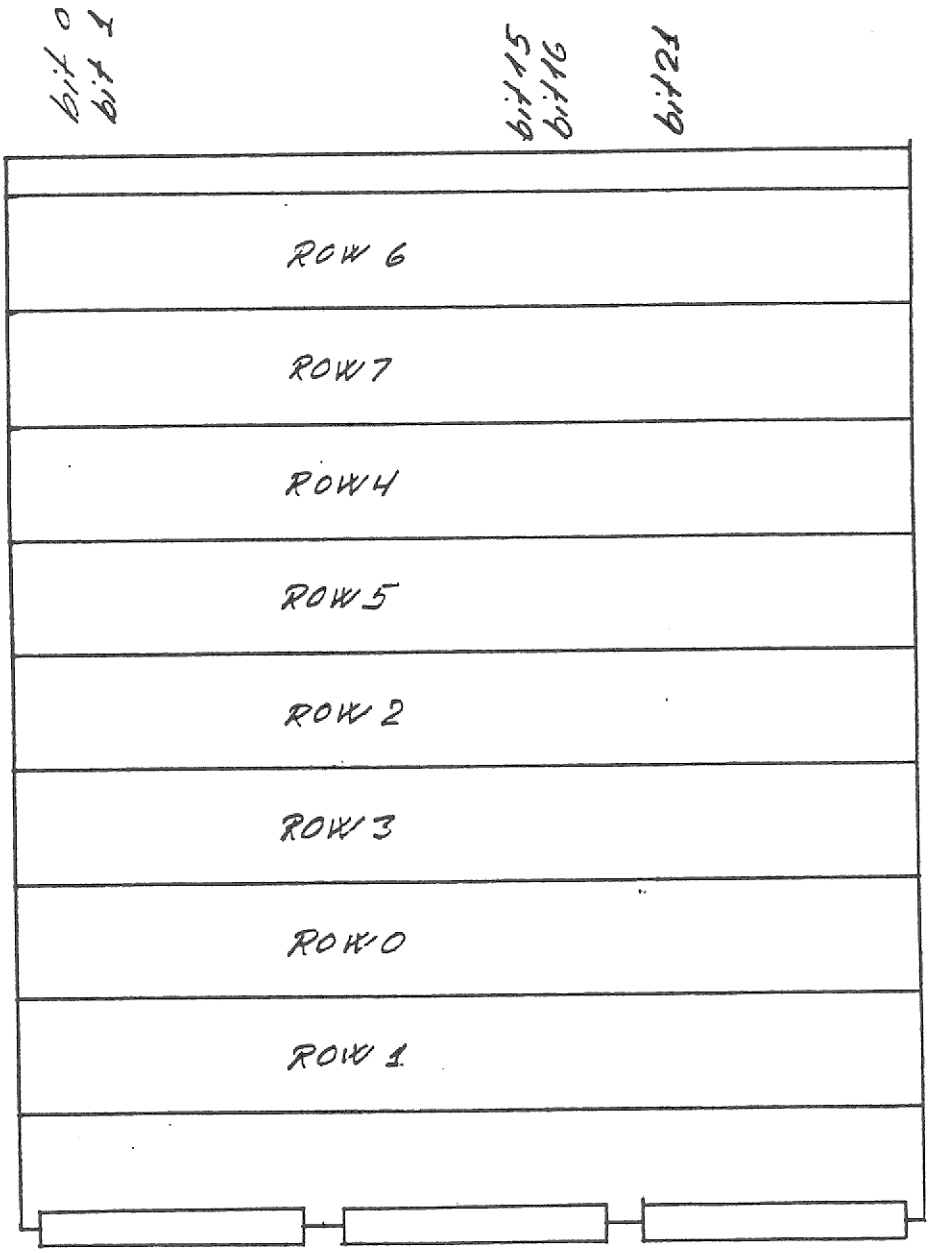
Acts in the same way as Control C.

Control S

When you press Control S the program will go in an idle state and wait. The selected test will be suspended. This feature is very useful when error messages starts to fill the screen. Then you can stop for a while to decode the message and then continue without having to restart program.

Control Q

Is used to resume program execution after a Control S has placed the program in the idle state. Only affects programs placed in idle state.


bit 0
bit 1
bit 15
bit 16
bit 21
ROW 6
ROW 7
ROW 4
ROW 5
ROW 2
ROW 3
ROW 0
ROW 1

Databit	Syndromebit
0	74
1	72
2	71
3	6C
4	6A
5	69
6	65
7	63
8	5C
9	5A
10	56
11	55
12	53
13	4E
14	4D
15	4B
16	7E
17	7D
18	7B
19	77
20	6F
21	5F