CMOS ULA

Testing

You have been allocated 9 hours of laboratory time in which to test your ULA designs. As a design team, the organization of your time is for the most part up to you. This document gives an outline of the available facilities and the tasks to be performed during this exercise.

Equipment

- Wafer Probe Station.
- SPARC Board computer.
- PC.
- D2 Interconnect Kit
- Design Exercise I.C. Kit
- Miscellaneous:
  - Experimentor boards
  - Logic Tutors
  - Standard Stores Components

  (as required)

Tasks

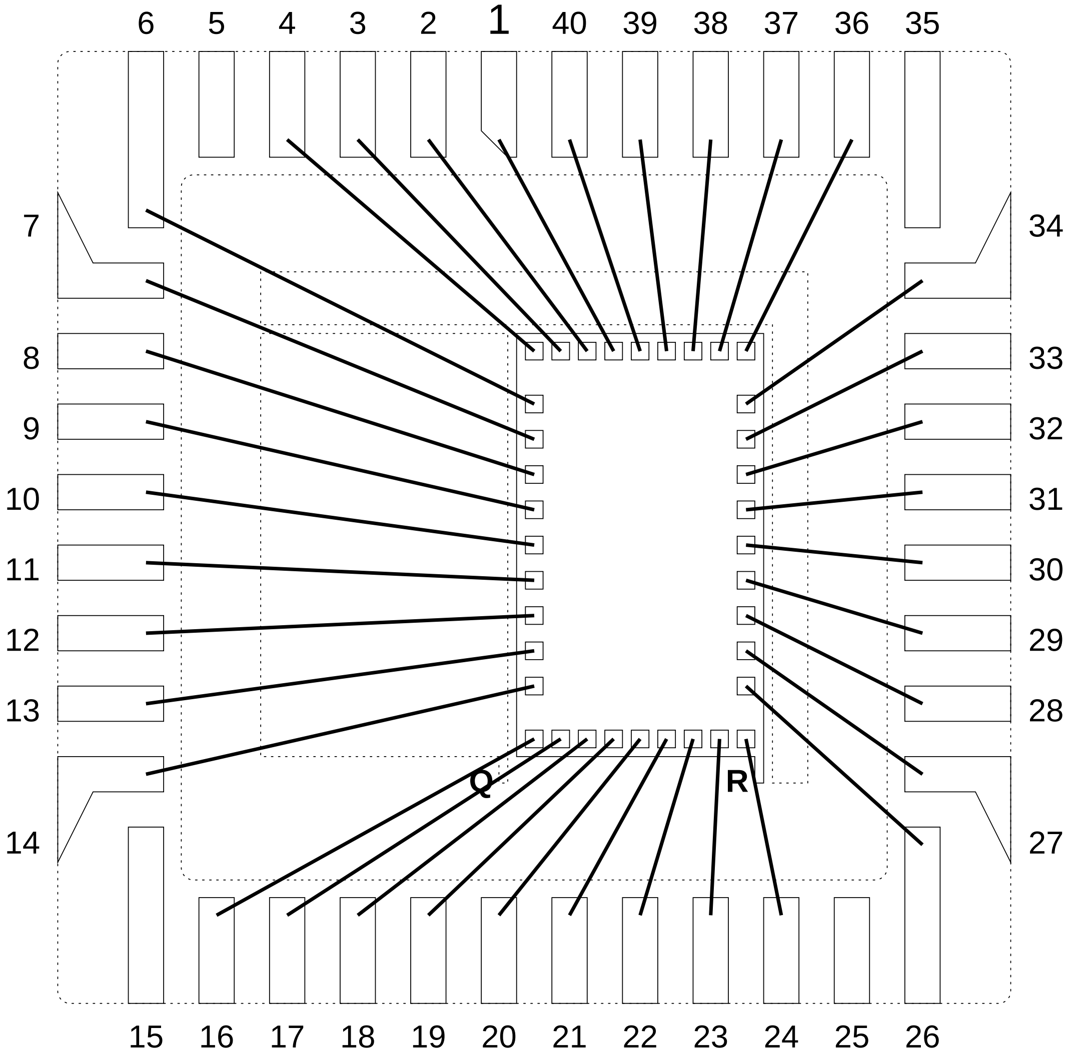
- Build Shaft Encoder Interface clone.
- Develop a full set of test vectors for your device.
- Analyse all your chips on at least one wafer.
- Build a full Serial Mouse Interface circuit to test the Shaft Encoder Interface unit.
- Demonstrate a functional Serial Mouse Interface circuit using the clone Shaft Encoder Interface unit.
- Demonstrate a functional Serial Mouse Interface circuit using the ULA Shaft Encoder Interface unit.

CMOS ULA

Wafer Probe Station.

There are five Wafer Probe Stations in the microprocessor applications laboratory. These stations allow testing of chips prior to dicing and bonding. In this way we avoid wasting effort in packaging faulty chips.
Each station should be fitted with a 36 pin probe card for making connections to ULA devices on a four inch silicon wafer. The probe card should be connected via ribbon cable to a 40 pin DIL header. The 40 pin DIL header is pin compatible with a bonded die.[1]

The bonding diagram for 36 pin ULA devices in 40 pin DIL packages is shown below:



Wafers, probe cards and probe stations are easily damaged through misuse.
Please seek advice and assistance from a demonstrator before using a probe station.

[1]Provided that the wafer is inserted into the prober with its flat side towards the operator - i.e. you.

CMOS ULA

SPARC Board Computer.

The SPARC Board (or RISC Experimenter Board) acts as the main processor for this exercise. It is interfaced to the device under test via its on-board Peripheral Interface & Timer (PIT) chips and a dedicated ULA Experimenter Board. It communicates with you, the user, via an RS232 connection to a PC.

- Chip Test Software.

  Each SPARC Board can run any of four EPROM based programs. By selecting **EPROM image 4** and pressing **Reset** you can select the I.C. test software required for this exercise.

- Interface Board.

  The SPARC Board should be connected via two ribbon cables to a ULA Experimenter Board. The 40 pin Zero Insertion Force socket on this board is designed to accept a ULA device in a 40 pin DIL package. It will also accept the 40 pin DIL header from the Wafer Probe Station.

  Power is supplied to the device under test by an external power supply via the power connections on the ULA Experimenter Board. Connections are also supplied on this board for the monitoring of current drawn by the device under test.

PC.

The networked PC next to your SPARC Board acts as a terminal and file store. The machines should be linked via an RS232 cable from the **COM2** port of the PC to **SERA** port of the SPARC Board.

- Terminal Program.

  The SPARC Board is accessed using the Windows Terminal Program set up for communication via COM2. Other parameters are; 19200 baud, 8 data bits, 1 stop bit, no parity and hardware flow control. This set-up is stored in a file called ULA.TRM in the windows directory.

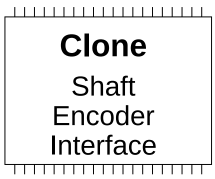
- File Store.

  When you log onto the PC with your own user i.d. and password you should gain access to your own private account space on drive **H:**. You can then use the **H:** disc space for storage of test vector files and results files.[2] Data is transferred to and from the PC using the **Receive Text File** and **Send Text File** facilities of the terminal program, or using the windows **Cut** and **Paste** facilities.

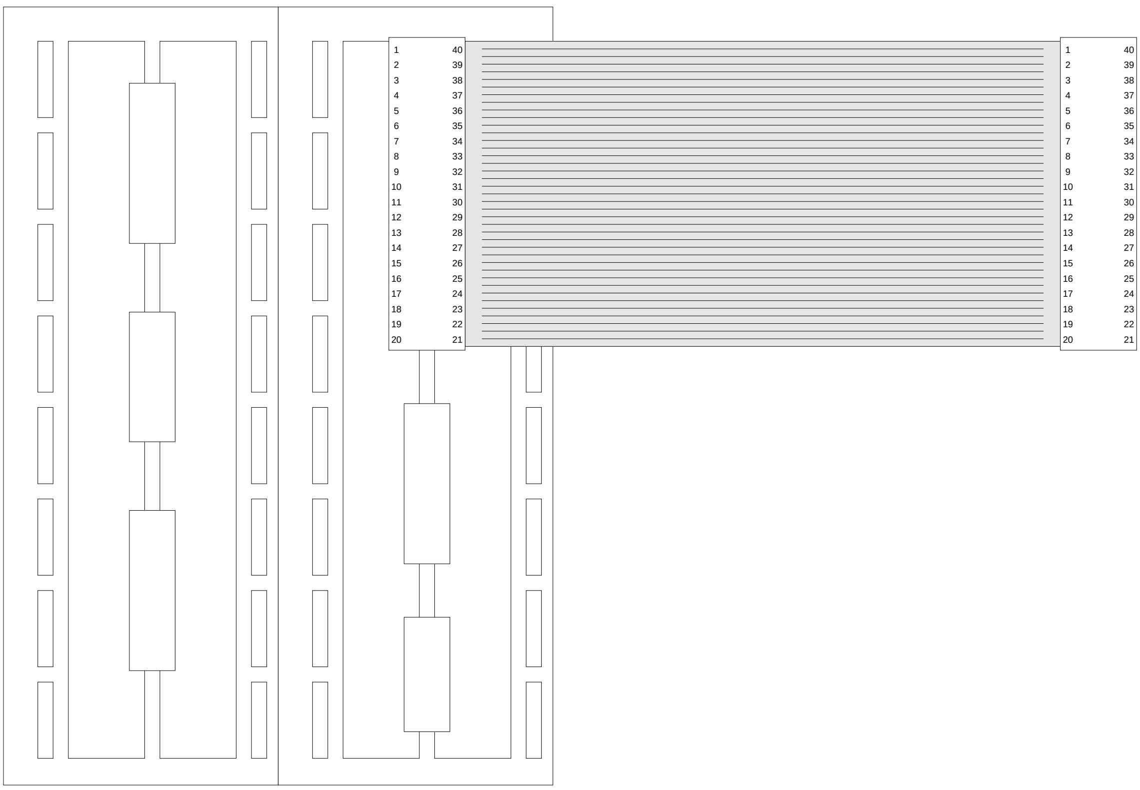
[2]Note that any files stored in a public area such as the **C:** drive may get overwritten or deleted between sessions or on reboot.

CMOS ULA

Build Shaft Encoder Interface clone.



This circuit should use standard components and customized PLDs to mimic the behaviour of your ULA based Shaft Encoder Interface unit.
The circuit can be built on one or a number of Experimentor breadboards. For connection to other circuits, you should be supplied with a ribbon cable with 40 pin DIL headers at each end. The circuit should be pin compatible with the ULA version, including all testability inputs and outputs. Thus any system developed to test the ULA can be verified using the clone circuit.



Develop a full set of test vectors for your device.

The SPARC Board computer allows you to create complex test vectors to check your circuit.

Test vectors are specified using a format statement followed by a number of vector specifications. The following test vectors might be used to check the RS flip-flop present on all chips:

```
"Test vectors for RS flip-flop"

     "TESTnS TESTnR TESTQ"
 f      26      27      24

 v      0       0       H
 v      1       0       L
 v      1       1       L
 v      0       1       H
 v      1       1       H
 v      1       0       L
 v      1       1       L
```

The `"......"` indicates a comment that is ignored by the software.

The `f` command specifies the format of the test vectors in terms of pin numbers.

Each `v` command specifies a set of inputs {`0,1,C,X`} to apply to the inputs and a set of results {`L,H,X`} to expect on the outputs.

Using `C` as an input results in a `010` clock sequence on the pin. It is often useful to specify a number of clock pulses explicitly before using the `C` macro.

Using `X` as an expected output allows you to ignore the value of one output while you are testing another output. Although `X` is supported as an input, there is little benefit in using it since it is always interpreted as a `0`.

You should aim to build up your test vectors until you have included tests for all parts of the circuit. The following vectors test an XOR gate as well as the RS flip-flop. The XOR gate has inputs at pins 32 and 33 and an output at pin 37.

```
"Combined test vectors"

 f       26 27  24    32 33   37

      "Test vectors for RS flip-flop"

 V001     0  0   H     0  0    X
 V002     1  0   L     0  0    X
 V003     1  1   L     0  0    X
 V004     0  1   H     0  0    X
 V005     1  1   H     0  0    X
 V006     1  0   L     0  0    X

      "Test vectors for XOR gate"

 V007     0  0   X     0  0    L
 V008     0  0   X     0  1    H
 V009     0  0   X     1  0    H
 V010     0  0   X     1  1    L
```

Vectors `V001-V006` test for the RS flip-flop as before and vectors `V007-V010` test for the XOR gate.

Note that the use of more formal, consecutively numbered, vectors helps to isolate errors while making it harder to insert vectors into the sequence.

A number of vector manipulation commands are available while using the chip test software. Useful commands include:

- `u` – updates the expected data based on the results of the last test. This is very useful for comparing one chip against another, or for comparing ULA chips against the clone.
- `v` (with no arguments) – prints the current vectors to the screen. The output from this command can be cut from the terminal window into an editor, allowing updated vectors to be stored and reused. Note that comments and spurious spaces are ignored by the software on input and thus are not returned by this command.

Type `help` at the command prompt in order to see a summary of these and other commands.

Note the importance of parallel development; the vectors can be used to debug the clone while the clone can be used to debug the vectors.

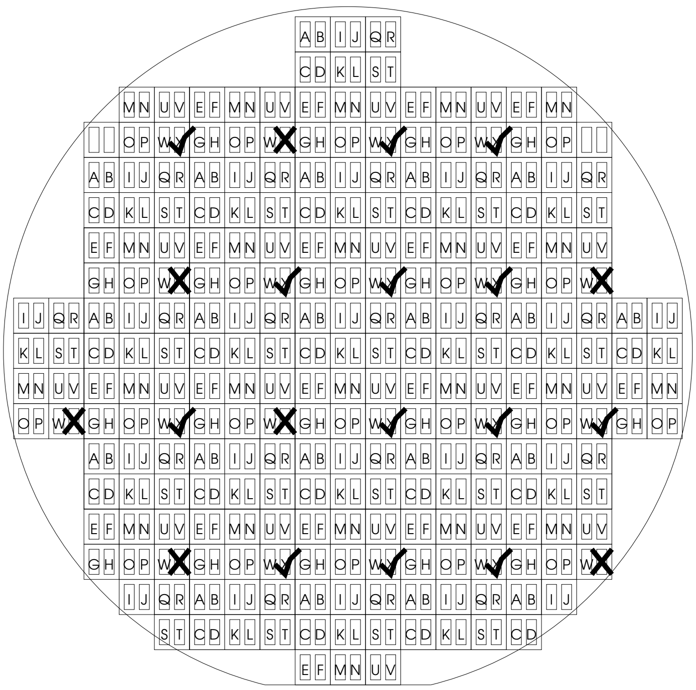
CMOS ULA

Analyse all your chips on at least one wafer.

Each wafer has over 450 ULA sites, of which approximately 20 are yours. Having located your ULA sites, identified by the team letter below and to the right of each ULA site, you should test each site for random process failures and systematic design faults.

Where a systematic failure is observed (such as an output which never changes on any of your ULA sites or one which consistently exhibits strange behaviour), you should attempt to identify its cause. Most systematic failures are caused by errors at the design or implementation stages[3]. In order to spot design errors, you will have to study your team's interim report and any netlist, icebox file or hard copy you have available for your design[4]. Of course the most useful way to spot a design error is through simulation of your design.
In the event of a design error you may wish to modify your test vectors to test for the circuit as it was implemented, as distinct from as you had hoped it was implemented.

Where random process failures are observed, such as an output which works on one site but not on another, you should use your test vectors to isolate particular circuits which are in error. It is then possible to build up a map of these process failures for the whole wafer, showing where failures are more frequent and indicating where correct chips can be found. This map should contain as much information as time allows. It is usual to record static current consumption on such a map as it provides an extra indication of the health of a circuit.



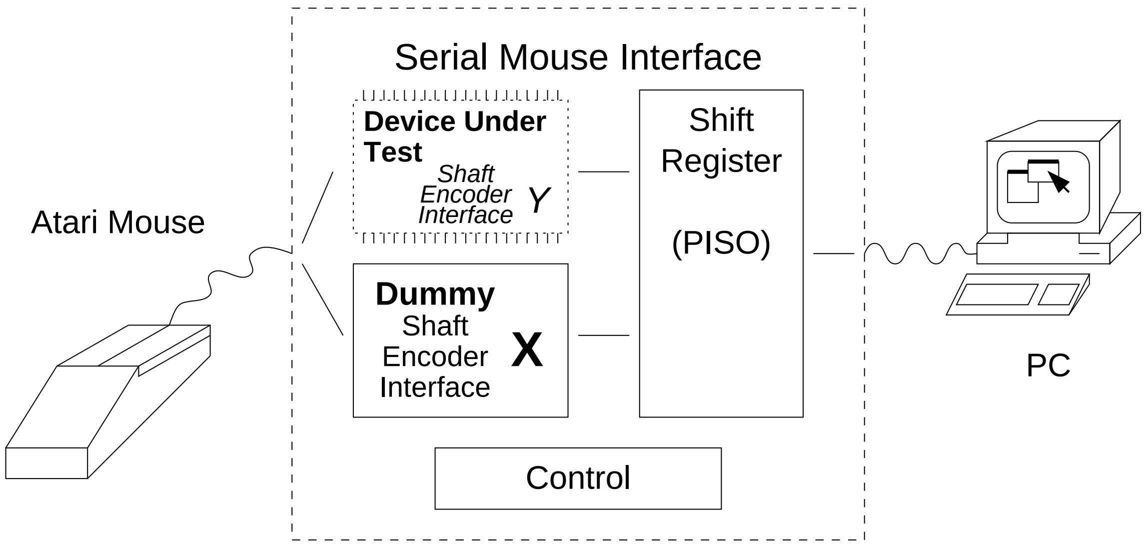
[3]Note – In rare cases, a reticle mask fault during fabrication may result in a systematic failure.
[4]Remember to bring all relevant documentation to the lab!

CMOS ULA

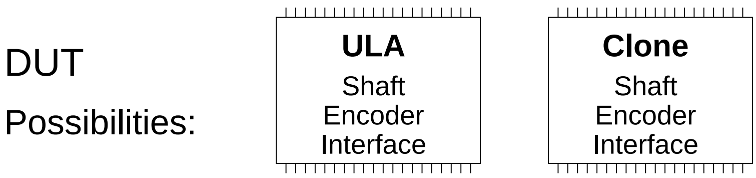
Build a full Serial Mouse Interface circuit to test the Shaft Encoder Interface unit.

The full circuit shown in outline below includes:

- Shift Register
- Control Unit
- Two Shaft Encoder Interfaces
  - Shaft Encoder Interface Y will be the DUT (device under test)
  - Shaft Encoder Interface X will be a dummy device with lesser functionality



The circuit should expect a 40 pin DIL package containing the ULA Shaft Encoder Interface unit that you have designed (this is the DUT). During the design and implementation stages of this exercise you should have separated your Shaft Encoder Interface into a number of testable sub-circuits. These sub-circuits are re-connected here in order to create a fully functional Serial Mouse Interface.
This circuit can be fully debugged using the Shaft Encoder Interface clone in place of the ULA version.



CMOS ULA

To simplify the task of building the full circuit, additional information is provided:

- A modified version of the original system diagram is included opposite.

  The modifications cope with a 32-bit shift register with active-low asynchronous reset and active-low parallel load enable signal rather than the **30**-bit shift register with active-low asynchronous **set** and active-**high** parallel load enable that was originally specified. The new 32-bit shift register can easily be built from the 74HC166 devices provided in the design exercise IC kit.

- A `control.pld` file is provided which can be used to program a PEEL18CV8 device to perform the functions of the Control unit.

- A `dummy.pld` file is provided which can be used to program a PEEL22CV10 device to perform the functions of a simple shaft encoder interface including a 3-bit counter. You can use this directly as the dummy shaft encoder interface required for a system test or you could use the code as the basis of a more advanced dummy with an increased number of counter bits.

- The serial mouse protocol is 3 bytes of RS232 data at 1200 baud. Each byte consists of a start bit (0), 7 data bits (LSB first) and 2 stop bits (11). Any idle time between messages is filled with more stop bits (111...1). The full system diagram shows the shift register inputs required to create the protocol. All that remains is to ensure a clock input as close to 1200 Hz as possible.

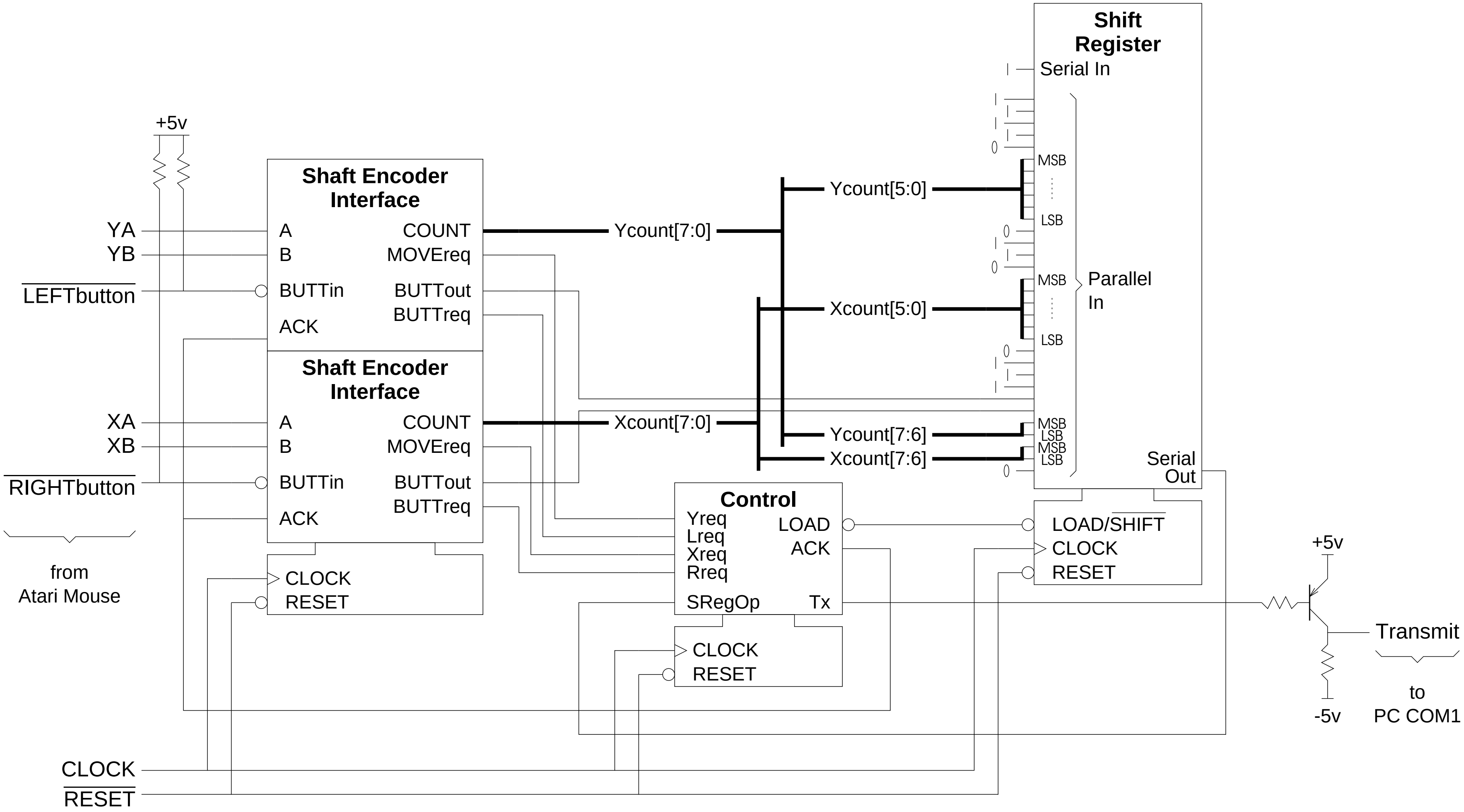
- Since none of the constructed shaft encoder interfaces support an eight bit counter, the counter values will have to be sign extended before wiring to the shift register. In order to give your mouse a faster screen movement (especially if you have only a few bits in your counter) you may like to pad the low order bits with zeros. e.g. a 3-bit counter might be padded with two zeros before sign extension to 8 bits: {COUNT2,COUNT2,COUNT2,COUNT2,COUNT1,COUNT0,0,0} — note that in this case COUNT2 is the sign bit.

Demonstrate a functional Serial Mouse Interface circuit using the clone Shaft Encoder Interface unit.

As with the test vectors you should use the clone to help in the debugging of your test circuit. A demonstration of the full system based on the clone is one of the deliverables for this exercise.

Demonstrate a functional Serial Mouse Interface circuit using the ULA Shaft Encoder Interface unit.

Having found a perfect ULA site (or one which is as near perfect as design and process failures allow) you should attempt to complete the exercise by connecting the ULA Shaft Encoder Interface to the full test circuit. You may need to bypass some ULA circuitry to make up for a less than perfect ULA sample.



+5v
Shaft Encoder Interface
YA
YB
LEFTbutton
A
B
BUTTin
ACK
COUNT
MOVEreq
BUTTout
BUTTreq
Shaft Encoder Interface
XA
XB
RIGHTbutton
A
B
BUTTin
ACK
COUNT
MOVEreq
BUTTout
BUTTreq
from Atari Mouse
CLOCK
RESET
Ycount[7:0]
Xcount[7:0]
Ycount[5:0]
Xcount[5:0]
Ycount[7:6]
Xcount[7:6]
Shift Register
Serial In
MSB
LSB
Parallel In
Serial Out
Control
Yreq
Lreq
Xreq
Rreq
SRegOp
LOAD
ACK
Tx
CLOCK
RESET
LOAD/SHIFT
CLOCK
RESET
+5v
-5v
Transmit
to PC COM1
CLOCK
RESET

Documentation

You are required to write a single group report on the work you have done for this exercise.

You should already have written an interim report on the work carried out last semester. Your new report should contain all the relevant information from the interim report. *Your full report should not assume that the reader has access to the interim report.*

Marks will be awarded for the following:[5]

- The design of the circuit and how well it meets the specification.

  Remember to include a clear and simple circuit diagram.

- The inclusion of testability at the design stage.

  Remember to include a diagram which shows all testability inputs and outputs as well as the system inputs and outputs.

- The implementation of the design onto silicon along with any measures taken to avoid errors.

  Remember to include a completed ULA template with gates and outputs uniquely identified.[6]

- Testing of the finished circuit.

  Include information on the various circuits constructed as well as the test vectors used.

- An analysis of all systematic failures.

  An updated circuit diagram may help here.

- A completed wafer map showing all random failures.

  Information too complex for a wafer map may be included in tabular form.

The report should clearly document all of the above, further marks will be available for the quality of the documentation. *No marks are available for returning to me, photocopies of documents I provided for you.*

An appendix to the report should indicate how the project was managed and how the tasks were divided amongst the members of the group.

[5] This list is not exhaustive but should give an indication of what is required.

[6] There is no point in identifying items here which can't be immediately identified in other diagrams.

Project Completion Form

Appended to this document there should be a project completion form. It provides a record of your achievements in the laboratory.
You should ensure that this form is completed and signed during the laboratory. The completed form must be appended to your report.

Time

The laboratory time is divided into two:

- 3 hour introductory session
- 6 hour main laboratory session

You should plan your work for the laboratory sessions and between laboratory sessions in order to make best use of the available resources.

- Wafers will only be available during the scheduled lab periods.

  During the introductory session you will share one probe station and wafer between two teams. During the main laboratory session you will have one probe station and one wafer per team. The reduced availability during the introductory session is for various logistical reasons, not least that we weren't sure that any wafers would be available by that time - the clean room staff have worked hard and provided us with 3 test wafers in time for the introductory sessions.

  In order to make most effective use of the wafers, you should attempt to get as much information as possible out of the wafer during the introductory session.

- Design Exercise I.C. Kits may be kept for the duration of the exercise.

  This allows you to build clone and test circuits outside of the scheduled sessions.

- D2 Interconnect Kits must be returned at the end of each lab session.

  In order to facilitate further testing, you may borrow these kits outside of your scheduled lab sessions provided that they are not required for other scheduled sessions and provided that you return them on the same morning/afternoon that you borrow them.

  A small number of clone cables may also be available for you to borrow when interconnect kits are in use.

Team Responsibility

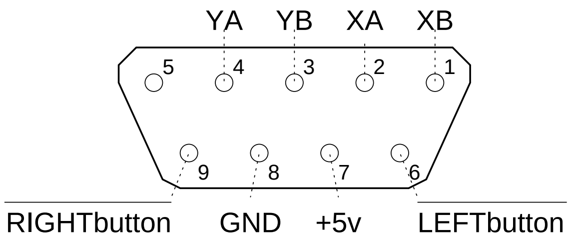
- Good teamwork and group management are essential for this project.
- The whole team is responsible for the whole project.

  Where a team member does not pull his or her weight, it is possible to flag this problem in the documentation and marks may be adjusted accordingly. Where one team member makes a mistake, the team is responsible for checking the work and correcting or compensating for this mistake.
- The whole team is responsible for the whole of the report.

  The report should read as a single document. You should allow time to read the whole document before it is submitted. A high degree of co-ordination is required in order to avoid contradictions and omissions.

Appendix — Resources

- D2 Interconnect Kit
  - ULA Experimenter Board + Cables – 2 × 26 pin IDC to 26 pin IDC
  - Cable for Prober – 40 pin IDC to 40 pin DIL
  - Wafer Tweezers
  - Cable for clone – 40 pin DIL to 40 pin DIL
  - Clone Rig (included though not required) – Experimentor board wired to 40 pin DIL
  - Connector for PC com1 port
  - Atari Mouse – contections as follows:



- Design Exercise I.C. Kit
  - 3 × PEEL18CV8
  - 2 × PEEL22CV10
  - 4 × 74HC166 – 8-bit PISO Shift Register