intellemetrics

Automated Carousel Test Glass Changer
User Manual

This documentation is provided as an instruction manual to

Intellemetrics customers and potential customers **only**.

Read this manual before you install and use the Automated Carousel Test Glass Changer.

Intellemetrics Global Limited
Suite 1014 Mile End Mill
Seedhill Road
Paisley
Renfrewshire
UK

Phone: +44 (0) 141 889 0700
Fax: +44 (0) 141 889 0707
Web: www.intellemetrics.com
e-mail: support@intellemetrics.com

Copyright Notice

TABLE OF CONTENTS

TABLE OF FIGURES

1. Nomenclature

Carousel:

The carousel is the circular holder for the test glasses.

Carousel Position:

The carousel positions are the physical positions as marked on the carousel. Intellemetrics manufactures carousels with various numbers of positions from 4 to 16 depending upon the geometry of the customer's chamber. The number of each position will be clearly marked on the carousel.

Carousel Rotation:

The carousel rotates in one direction only, it cannot be reversed. Therefore to go back one Carousel Position requires almost a complete rotation.

Test Glass:

A test glass is the piece that is being optically monitored. Typically these are glass discs (19.8mm diameter x 3mm thick), although other sizes and materials can be used.

Test Glass Number:

The test glass number refers to the number used in the optical monitoring scheme. This is different from the Carousel Position. For instance, Test Glass Number 1 can be in Carousel Position 4.

Logic '1' or '0':

In this document a logical '0' is no volts or current on a signal while a logical '1' is an adequate voltage or current on a signal.

2. Power

The Test Glass Changer is powered by 12V DC via a two pin power connector, Lemo Type FGG.0B.302.CLAD42, with pin 2 +12V and pin 1 ground or 0V. The supply must be between 13.2V and 11.0V for guaranteed reliable operation. The supply must be able to provide 0.7A with a ripple of less than 250mV.

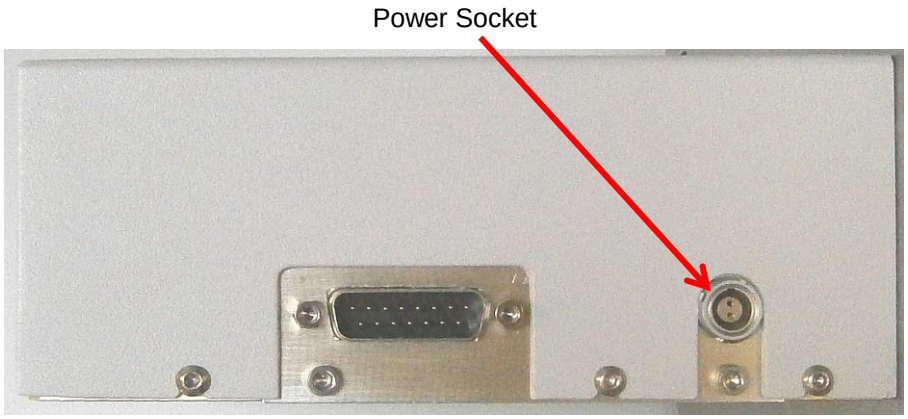

Figure 1: Test Glass Changer Power Socket

3. Environment

While the electronics have been designed using extended temperature components they will not survive extended bake outs at elevated temperatures if powered up. Therefore the power should not be turned on if the temperature of the Test Glass Changer metalwork is above 50^{o}C assuming the electronics are being operated at +5V.

Equally the power should not be turned on if the relative humidity is above 90% or is condensing.

4. Inputs

The inputs are optically isolated from the internal electronics. There is a common connection for all the inputs on pin 12 of the 15 pin 'D' type connector. Each input can be driven with either a positive or negative signal with respect to the common pin.

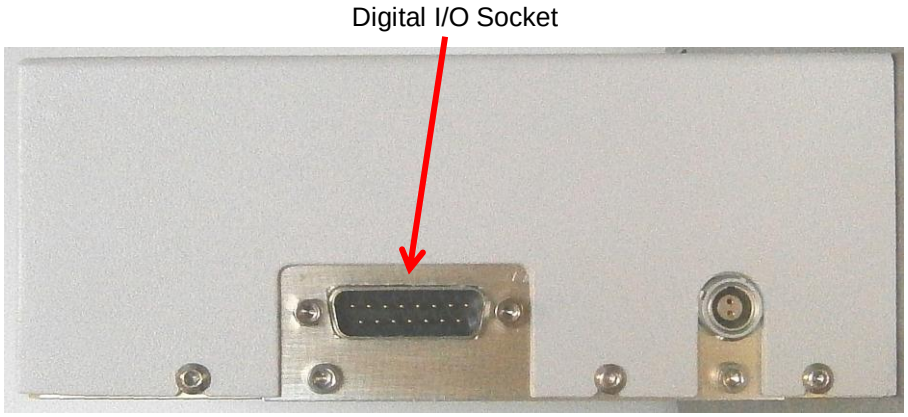

Figure 2: Inputs and Outputs to the Test Glass Changer are via the 15 way 'D' type connector.

The voltage level on an input for '0' is between +/- 1.2V while a '1' is +5 to +24V or –5 to –24V. The current required is between 3.8 and 22.8mA. Refer to Appendix B for further details.

The pin out is: -

Pin Number	Signal Name
1	New Carousel Position 3
9	New Carousel Position 2
2	New Carousel Position 1
10	New Carousel Position 0
3	Load New Carousel Position
11	Enable Motor
4	Reset
12	Common

The ***New Carousel Position 0*** signal is the least significant bit of a four bit binary address while the ***New Carousel Position 3*** is the most significant bit. The first carousel position is therefore address 0000 while the eighth carousel position is address 0111 (binary 7). For simplicity it is easier to refer to the carousel positions numbered from 0 to a maximum of 15, depending on the number available in each system. Refer to Appendix A for an address table.

The ***New Carousel Position*** signals are ignored until ***Load New Carousel Position*** is set to a '1'. The Test Glass Changer will try to start rotating if a valid address has been provided, but the address is not stored until ***Load New Carousel Position*** is returned to a '0'. This means that if the ***New Carousel Position*** signals change while ***Load New Carousel Position*** is '1' only the address when ***New Carousel Position*** returns to '0' is stored. Therefore there is no tight timing requirement to be met and the system connected to the Test Glass Changer can change all these signals at the same time without causing any timing glitches provided that the ***New Carousel Position*** signals are not changed until after ***Load New Carousel Position*** has gone to '0' and need not change until a new address is set the next time ***Load New Carousel Position*** is being set to a '1'. If the same address as the Test Glass Changer is currently stopped at is requested again the request will be ignored.

Enable Motor is used to ensure that the Test Glass Changer can only rotate when this signal is set to '1'. This should be used as a safety feature to protect an operator changing the test glasses. It is probably best used with a door safety interlock or pump down interlock, when the system is in a safe condition ***Enable Motor*** can be left in the '1' state.

When the Test Glass Changer is powered up it goes to the ***Reset*** state. The ***Reset*** signal will place the Test Glass Changer into the ***Reset*** state at any time if set to '1'. The Test Glass Changer will not start executing the ***Reset*** sequence until the signal is returned to a '0' level. ***Enable Motor*** must be in the '1' state for the motor to rotate to allow the ***Reset*** sequence to complete. If ***Reset*** and ***Enable Motor*** are both held in a '1' state the Test Glass Changer continually rotates.

The ***Reset*** sequence can take up to two full revolutions of the Test Glass Changer to complete. For the slowest motor speed (1.5rpm) this is a maximum of 80 seconds. On completion the Test Glass Changer will be on the zero position.

5. Outputs

Once again the outputs are optically isolated with the output being driven by a Darlington transistor pair with a reverse diode fitted to protect the transistor from any reverse voltage due to switching an inductive load. All the outputs have a common signal connected to the emitters of the transistors and therefore the outputs can only switch a positive voltage.

The pin out is: -

Pin Number	Signal Name
5	Common
13	Current Carousel Position 3
6	Current Carousel Position 2
14	Current Carousel Position 1
7	Current Carousel Position 0
15	Carousel Position Valid
8	Error

The maximum voltage that can be applied between ***Common*** and any output is 24V. The transistor can sink between 1 and 30mA with a voltage of 1.2V left between ***Common*** and the output. Typically a pull up resistor would be fitted to the external supply and the output. The voltage on the output would be high when the transistor was not conducting and low when it was. Refer to Appendix B for more information.

Current Carousel Position 3 to ***Current Carousel Position 0*** is the binary number of the current carousel position if ***Carousel Position Valid*** is a logic '1' which means that the transistor is off and not conducting any current. ***Carousel Position Valid*** will only be logic '1' when the motor is not rotating. Therefore the input ***Enable Motor*** does not change the state of this output. ***Carousel Position Valid*** will also not be valid until the end of the ***Reset*** state has been reached when ***Current Carousel Position*** will read an all '0' state signifying that the Test Glass Changer has stopped at the first carousel position. Remember that ***Carousel Position Valid*** will change state to logic '0' as soon as the Test Glass Changer is requested to move to

another carousel position by setting ***Load New Carousel Position*** to logic '1' even if the ***Enable Motor*** input will stop the motor from rotating.

The ***Error*** signal will go to logic '1' to flag up two possible errors. Firstly if the motor drive chip overheats and stops the motor, the drive chip will automatically reset itself until the output drivers have reached their operational temperature again and the motor will start to rotate, the Test Glass Changer will continue as if no error has happened. Secondly if an invalid ***New Carousel Position*** is requested with ***Load New Carousel Position*** at logic '1' the ***Error*** signal will be set and the Test Glass Changer will not execute the command.

6. Suggested Programming Sequences

There are example Visual Basic 2010 routines from our diagnostics available in Appendix C.

6.1. At Initial Power Up

1. Ensure the power to the Test Glass Changer is on.
2. Set ***Enable Motor*** to logic '1' when safe to do so.
3. Monitor ***Carousel Position Valid*** going to logic '1'.
4. Also monitor ***Error***, this would only be set if the motor drive chip overheats.
5. When ***Carousel Position Valid*** is '1' set ***Enable Motor*** to '0'.
6. Starting at address 0 increment through the ***New Carousel Positions***, leaving ***Load New Carousel Position*** at '1' while monitoring ***Error***. This can be done at 1ms per address maximum.
7. If ***Error*** is set then you have ***Current New Carousel Position*** -1 carousel positions in the Test Glass Changer.
8. If ***Error*** is never set then you have 16 carousel positions in the Test Glass Changer.
9. Set ***New Carousel Position*** to zero and drop ***Load New Carousel Position*** to '0', set ***Enable Motor*** to '1' to enable the motor drive.

6.2. Selecting a Carousel Position

1. Ensure ***Enable Motor*** is '1' and ***Carousel Position Valid*** is '1' and ***Error*** is '0'.
2. Set ***New Carousel Position*** to values in the range 3 to 0 and ***Load New Carousel Position*** to '1' which can all be done at the same time.
3. Check that ***Carousel Position Valid*** has changed to a '0' and ***Error*** is '0'. This could take 1ms from step 2.
4. Set ***Load New Carousel Position*** to a '0' state to store the new Carousel Position.
5. Check ***Carousel Position Valid*** until it changes to '1'. Remember that it takes rpm of motor times 60 divided by the number of carousel positions in seconds between each carousel position that you are rotating past. Therefore checking status once per second should be the maximum that you do; a slower rate will be more than adequate for most installations. You should also check ***Error*** is '0' as well, refer to section **'Handling an Error'** if it is a '1'.
6. Read in ***Current Carousel Position*** 3 to 0 and check that the Test Glass Changer has stopped at the correct carousel position.
7. If the Test Glass Changer has stopped at the wrong carousel position twice in succession then go to **'How to Reset the Test Glass Changer'**. If this does not fix the problem then the fault must be investigated.

6.3. Handling an Error

1. If ***Error*** is '1' then there are three possibilities, one being outside the Test Glass Changers control.
2. Set ***Enable Motor*** to '0' to stop rotation.
3. Set ***New Carousel Position*** 3 to 0 to address zero and set ***Load New Carousel Position*** to '1'.
4. If ***Error*** is still '1' then the motor drive chip has overheated and will reset itself when the temperature has dropped. Keep poling ***Error*** every few seconds until you have at least two consecutive '0' states read.
5. Set ***New Carousel Position*** 3 to 0 back to their original address, afterwards set ***Load New Carousel Position*** to '0' and ***Enable Motor*** back to '1' and jump to the end.
6. If ***Error*** was '0' at step 4 then there is an addressing problem where the carousel position that has been requested is greater than the number of carousel positions the Test Glass Changer believes are present.

7. Check the address requested is not greater than the number of carousel positions found in **'At Initial Power Up'**. If it is then send a valid address following **'Selecting a Carousel Position'**.
8. If the address was not greater than the number of carousel positions then the Test Glass Changer has either had a power glitch, excessive EMC interference, exceeded its environmental specification or this interfacing specification has not been met. Follow the section **'How to Reset the Test Glass Changer'** and have the causes above investigated.

6.4. How to Reset the Test Glass Changer

1. Set ***Enable Motor*** and ***Load New Carousel Position*** to '0'.
2. Set ***Reset*** to '1' for at least 100ms.
3. Set ***Reset*** to '0'.
4. Follow **'At Initial Power Up'** from step 2.

6.5. Removing and replacing the Carousel

1. The safest option is to power off the Test Glass Changer. When power is restored follow **'At Initial Power Up'**.
2. If the chamber is only being opened for a short time and it is preferred to leave the power on then set ***Enable Motor*** to '0' to prevent the motor from moving while the operator is changing the Carousel.
3. When the chamber is safe again, the preferred route is to follow **'How to Reset the Test Glass Changer'**; alternatively you can continue with the next step.
4. Check ***Carousel Position Valid*** is still '1' and ***Current Carousel Position*** 3 to 0 have not changed, if this is the case then set ***Enable Motor*** to '1' and continue as the Test Glass Changer did not move during the Carousel replacement.
5. If the check in 4 above is not true then go to **'How to Reset the Test Glass Changer'.**

The above sequences should enable all of the required operations on the Test Glass Changer to be programmed for reliable operation.

7. Appendix A – Binary Addresses & Carousel Positions

A table of binary addresses and carousel position numbers.

Carousel Position	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Bits 3 to 0 equate to ***New Carousel Position*** 3 to 0 and ***Current Carousel Position*** 3 to 0.

8. Appendix B – Input / Output Circuits

Here are the details of the input and output optically isolated circuits used in the Test Glass Changer.

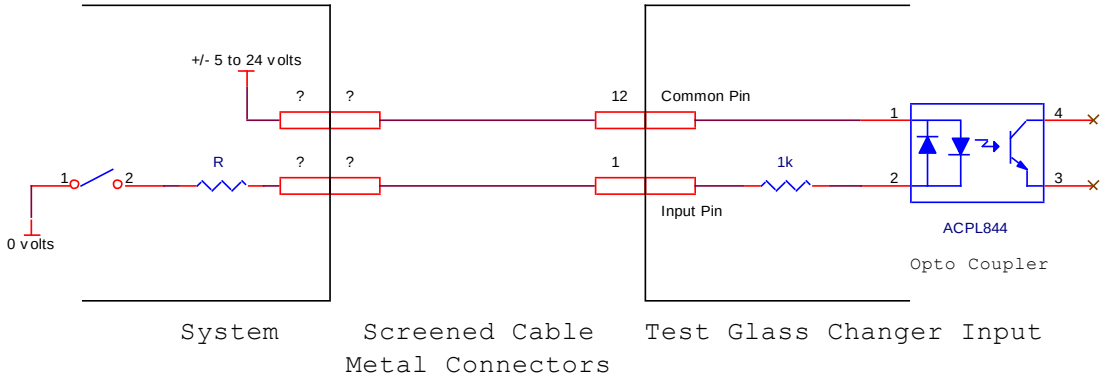

Figure 3 and Figure 4 are illustrations of two different options for the inputs to the Test Glass Changer. The switch shown is a representation of a systems output, either a relay contact, a single transistor or Darlington driver.

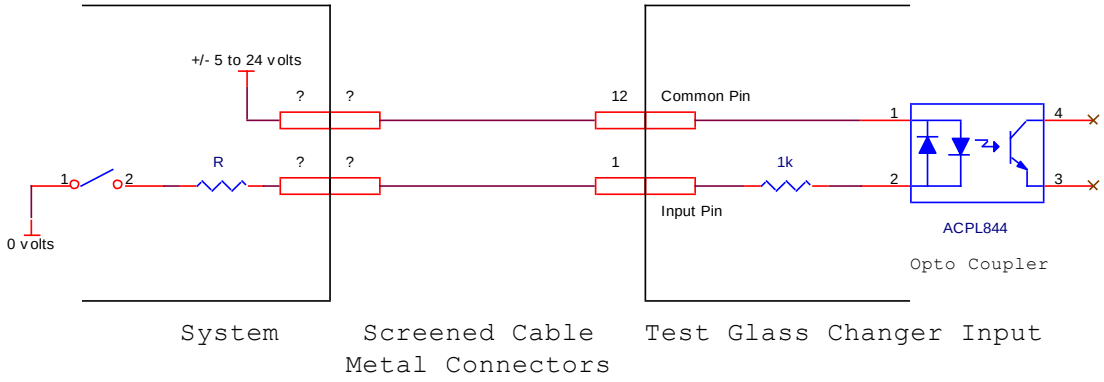

Figure 3: Test Glass Changer Inputs Option 1.

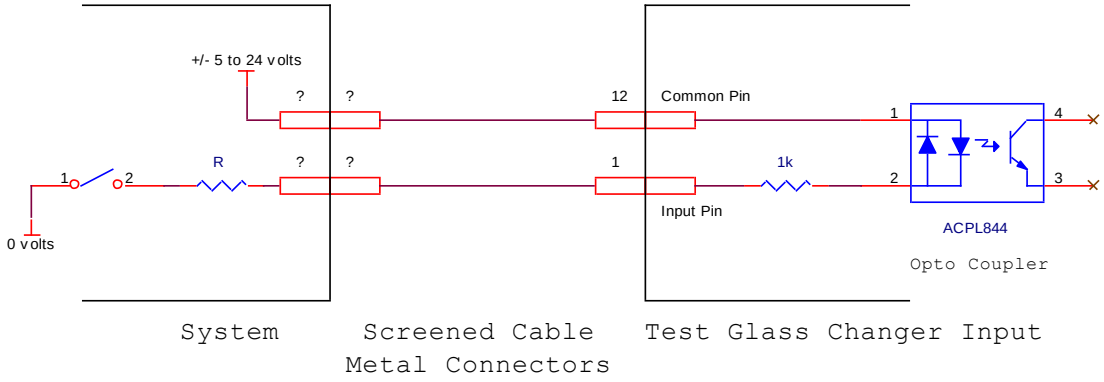

Figure 3 is more energy efficient and is the preferred method if the system output is a Darlington driver and there is more than 7V supply as the saturated voltage across the Darlington will not affect the current available through the optical coupler. The dissipation in the 1kΩ resistor should be calculated and kept below 300mW. For example R could be 1kΩ if 24V is being used while still allowing over 10mA to flow through the optical coupler diode. If the available voltage is 5 to 7V then the circuit in Figure 4 must be used.

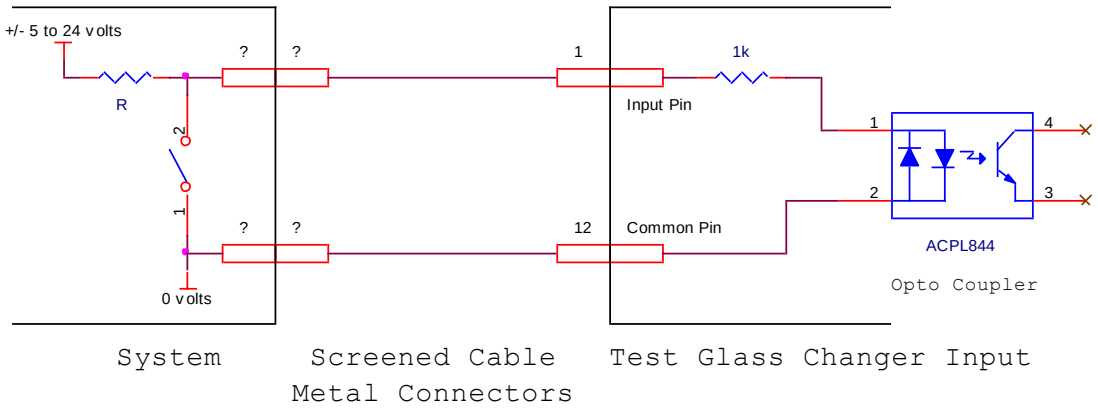

Figure 4: Test Glass Changer Inputs Option 2.

Figure 4 needs to be used if there is any voltage drop across the system switch and a voltage of 5 to 7V is to be used. R will be a low value resistor, 100Ω at 5V, and rated at more than 250mW. This circuit also inverts the sense of the signal as driver on means the input optical coupler has no current, and off lets the current flow.

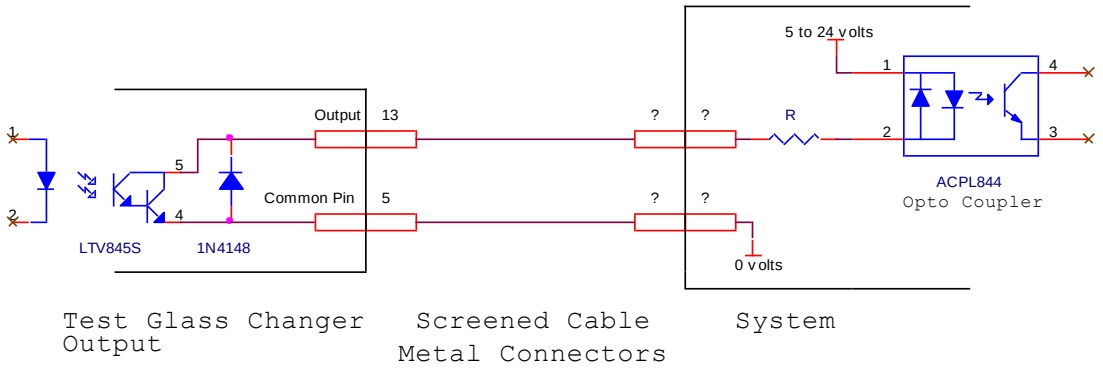

Figure 5: Test Glass Changer Outputs Option 1

Figure 5 is similar to the input circuit in Figure 3. R must be calculated to allow sufficient current to flow in the input diode for the optical coupler to change state. This circuit will work for 5 volts as long as R is kept low enough, 10mA equates to around 220Ω.

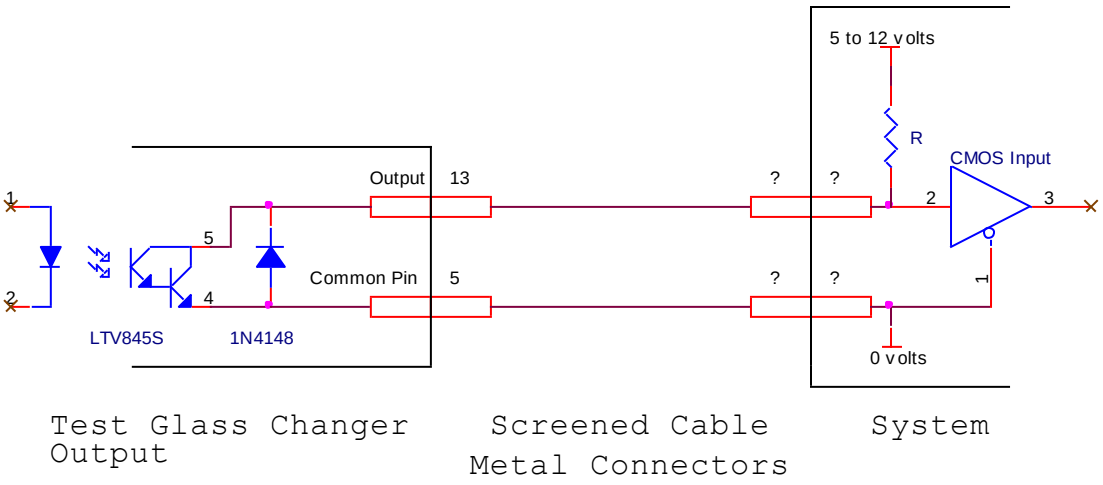

Figure 6: Test Glass Changer Outputs Option 2

Figure 6 is an example of the outputs driving a logic gate input. The saturated on voltage across pins 5 and 13 is 1.2V and this will produce a low level input to most logic families. R must be fitted to ensure that when the output is off the input to the logic gate will be pulled to logic ‘1’ and also to provide an adequate current to be switched by the output.

9. Appendix C – Programming Examples

Here are simplified examples of programming the automated carousel test glass changer. They have been taken from our development and test routines which were written using Microsoft Visual Basic 2010.

A few items to note are :-

- Common.Reset The Common. Prefix refers to a class library where all the bit identities are defined.
- OutputByte is a routine which takes the signals defined in Common and maps them to a real output port.
- InputByte is a routine which does the same for the input signals.
- Waitmsec(100) is a routine that waits for the number of specified msec in this case 100.
- It should be noted that there is no error checking in the routines which should be added for production code although there is an example of identifying what caused an error.

9.1. Reset

The reset routine will force a ***Reset*** at any time. If the power has just been applied then it is not necessary to use the ***Reset*** signal at all but by doing so the routine can be used at any time.

```
    Private Sub reset()
        'this routine will reset the chip changer and find out how many carousel
        'positions there are and display them
        'force the reset to start
        OutputByte(Common.Reset Or Common.Motor_Enable)
        'pause a moment, the carousel will rotate continuously until the reset signal is
removed
        waitmsec(100)
        'remove the reset signal but leave the motor rotating
        OutputByte(Common.Motor_Enable)
        'wait for up to 2 revolutions until we arrive at a valid position 0
        Do
        Loop Until InputByte() And Common.Carousel_Position_Valid
        'find the number of carousel positions and display them
        TextBox4.Text = NumberOfChips()
    End Sub
```

9.2. Function NumberOfChips

This function will step through all the carousel addresses from 15 downwards looking at the ***Error*** signal which will drop when we arrive at a valid carousel address. It does this with the ***Motor Enable*** signal disabling rotation. This can only be used after a valid ***Reset*** sequence has been run when it can then be used as often as required. In a production environment where the number of carousel positions are known this can be used to verify that the Test Glass Changer is working correctly.

```
    Private Function NumberOfChips() As UInteger
        'This routine finds out how many carousel positions there are
        'without moving the carousel
        Dim byte4 As Byte = Common.Load_New_Carousel_Position + Common.Address_Mask + 1 'the
+1 is subtracted
        'clear the outputs, this will also clear Motor Enable so the carousel
        'will not rotate
        OutputByte(0)
        'just pause
        waitmsec(100)
        'count down from 15 until there is no error
```

```
        Do
            byte4 = byte4 - 1 'this is where the byte4 +1 is removed first time through the
loop
            OutputByte(byte4)
            waitmsec(100) 'remember to pause
        Loop While InputByte() And Common.Error_Mask
        'clear the outputs
        OutputByte(0)
        'this is the number of carousel positions starting from 0
        Common.NumOfChips = byte4 And Common.Address_Mask
        'put the number of positions into human readable form starting from 1
        NumberOfChips = CUInt(Common.NumOfChips + 1)
    End Function
```

9.3. Routine Position

The routine will position the carousel to the selected position starting at position 1. It first checks, using CheckInitialised, to make sure that the Chip Changer has run the ***Reset*** sequence, if not it does so as part of routine CheckInitialised. It then checks that a valid position is being requested otherwise it exits back to the requestor. If all is correct it then moves to the requested position.

```
    Private Sub position(ByVal chip_position As Integer)
        'This routine positions the carousel to the selected position
        'Check we have had a reset and are in a valid position
        CheckInitialised()
        'is an invalid carousel position being requested
        If (chip_position - 1) > Common.NumOfChips Then
            Exit Sub
            'are we being asked to go to the same position
        ElseIf (chip_position - 1) = InputByte() And Common.Address_Mask Then
            Exit Sub
        End If
        'start moving
        OutputByte((chip_position - 1) Or Common.Load_New_Carousel_Position Or
Common.Motor_Enable)
        'wait
        waitmsec(100)
        'drop Load New Carousel Position
        OutputByte((chip_position - 1) Or Common.Motor_Enable)
        'wait after an output
        waitmsec(100)
        'just wait until we arrive at our position
        Do
        Loop Until InputByte() And Common.Carousel_Position_Valid
        'clear the outputs
        OutputByte(0)
    End Sub
```

9.4. Routine CheckInitialised

Here is a reliable way to tell if the ***Reset*** sequence has run. On power up or ***Reset*** the ***Current Carousel Position 3 to 0*** bits will show all '1' giving address 15 but there will be no ***Carousel Position Valid*** signal. If this is the case then checking address 1, which will always be valid after the ***Reset*** sequence as there will always be more than two carousel positions, tells us what we need to know. Do not use address 0 as this will show as valid, no ***Error*** signal, after power up even if the ***Reset*** sequence has not run or been completed. In a production environment you will always know how many carousel positions there are which also helps with checking.

```
Private Sub CheckInitialised()
    'Check to see if the chip changer has been reset
    'and therefore should be in a valid position
    Dim byte9, byte10 As Byte

    OutputByte(0)
    waitmsec(100)
    byte10 = InputByte()
    byte9 = byte10 And Common.Carousel_Position_Valid
    If (byte10 And Common.Address_Mask = 15) And (byte9 = 0) Then
        'this will address position 1 without rotation
        OutputByte(1 + Common.Load_New_Carousel_Position)
        waitmsec(100)
        'check for error
        If InputByte() And Common.Error_Mask Then
            'not run reset yet or power glitch
            reset()
            Exit Sub
        End If
    End If
    'check if we already know the number of chips
    If Common.NumOfChips = 0 Then
        'if not fetch them and display
        TextBox4.Text = NumberOfChips()
    End If
End Sub
```

9.5. Routine identify_error

This is an example of how to identify the different types of errors which could occur. In a production environment any unexpected occurrence of the ***Error*** signal would call a routine like this. The error type would be logged and the necessary actions taken to recover.

```
Private Sub identify_error()
    'This routine follows the steps to identify if an error
    'is due to trying to select a carousel position greater
    'than the number present or due to the motor drive chip
    'overheating
    'check for error
    If (InputByte() And Common.Error_Mask) = 0 Then
        ErrorMessage.Items.Add("No error found")
        Exit Sub
    End If
    'disable motor, load address 0
    OutputByte(Common.Load_New_Carousel_Position)
    waitmsec(100)
    If (InputByte() And Common.Error_Mask) = Common.Error_Mask Then
        ErrorMessage.Items.Add("There has been a motor drive fault")
        'keep checking until error clears
        Do While (InputByte() And Common.Error_Mask)
            waitmsec(500)
        Loop
        'wait, then
        waitmsec(500)
        'check again to make sure
        Do While (InputByte() And Common.Error_Mask)
            waitmsec(500)
        Loop
        Exit Sub
```

```
            'check to see if initialised
        ElseIf Common.NumOfChips = 0 Then
            reset()
            ErrorMessage.Items.Add("The carousel had not been initialised")
        Else
            OutputByte(Common.Load_New_Carousel_Position + Common.NumOfChips)
            waitmsec(100)
            If (InputByte() And Common.Error_Mask) = 0 Then
                ErrorMessage.Items.Add("A non existent position had been requested")
            Else
                ErrorMessage.Items.Add("There has been a glitch, Reset")
            End If
        End If
        OutputByte(0)
    End Sub
```