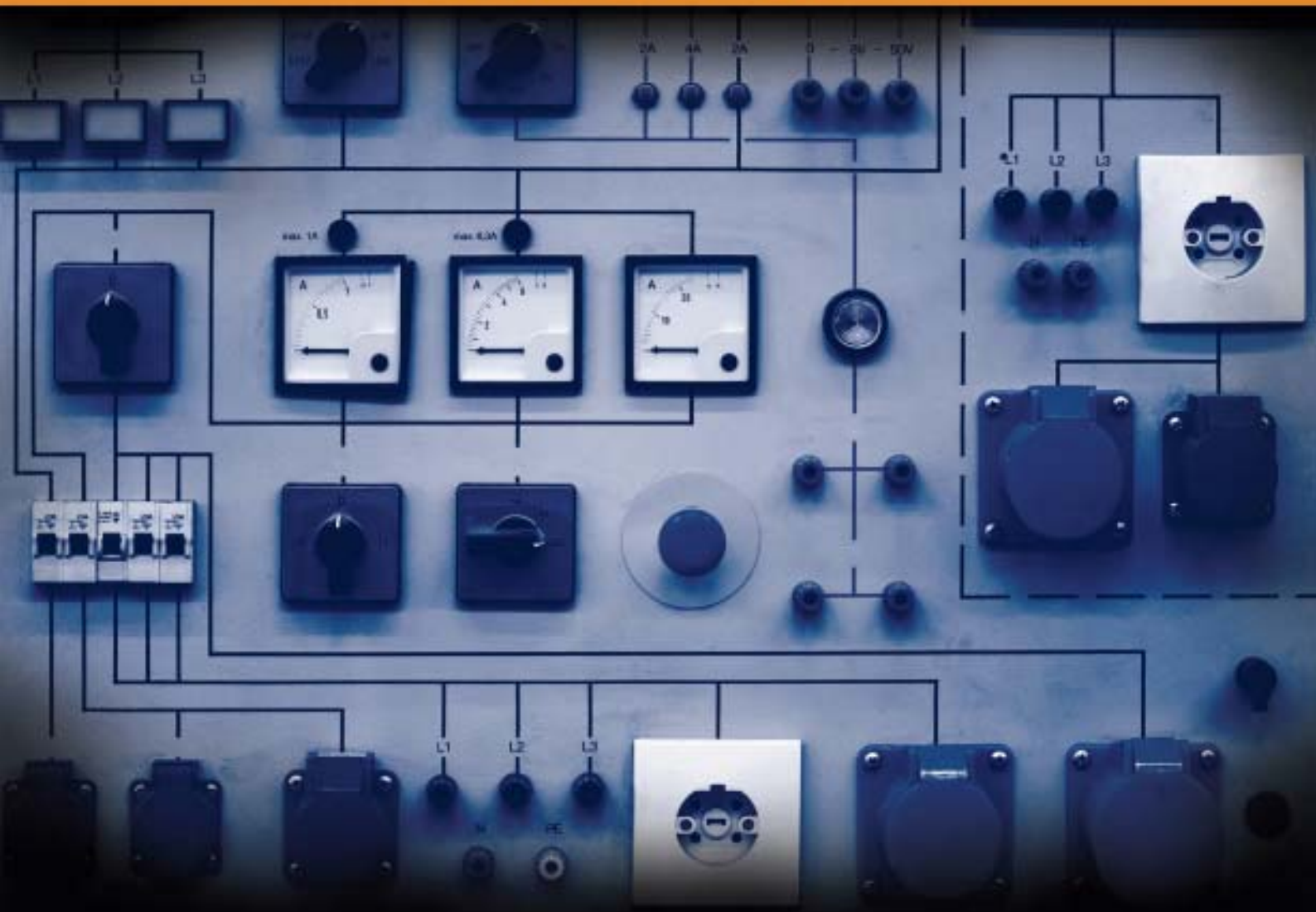
eBLOCKS®

LEARN·DESIGN·BUILD

Actuators Training Panel



TEFWS

MATRIX

www.matrixmultimedia.com

Contents

Sensors and Actuators

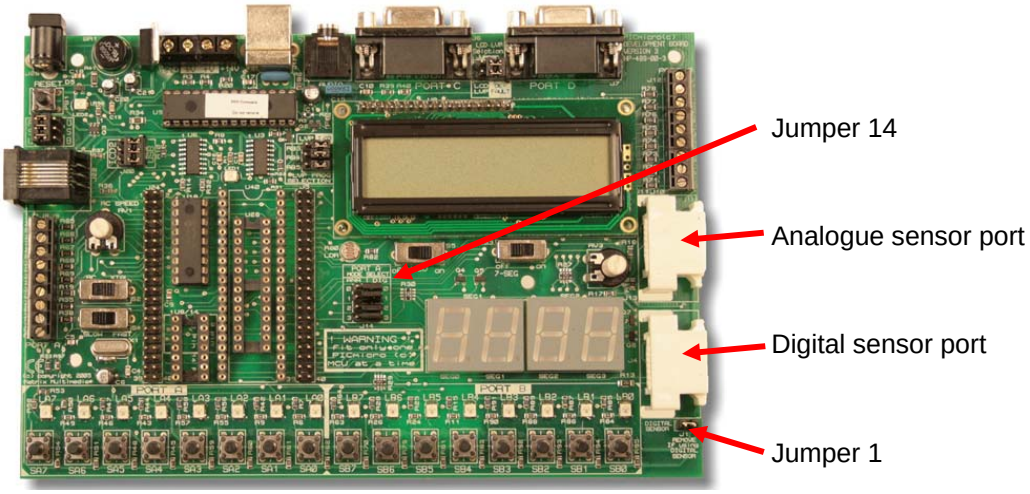
The hardware	3
Testing the hardware	7
Worksheet 1 - Driving the DC motor	8
Worksheet 2 - Driving the stepper motor	10
Worksheet 3 - Driving the servo motor	12
Worksheet 4 - Measuring heart-rate	14
Worksheet 5 - Measuring temperature	16
Worksheet 6 - Timing using the photo-gate	18
Worksheet 7 - Measuring motion	20
Tutor’s notes	22

The hardware

Sensors and Actuators

Using the PICmicro Development Board Version 3

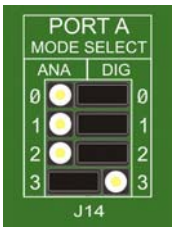
- programs a range of 8, 14, 18, 28 and 40 pin PICmicro® microcontrollers;
- programs via the USB port;
- powered from USB or external power supply such as HP5328 adjustable power supply (recommended.)



3a

External analogue sensor:

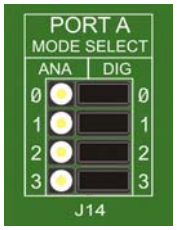
- requires a PICmicro chip with analogue capabilities, such as the PIC16F874 or PIC16F877;
- is plugged into the analogue sensor port (upper white connector on the right-hand side of the PICmicro development board,);
- requires corresponding part of jumper 14 - the Port A mode selector - set to analogue (ANA).
  (The diagram shows the setting for an analogue sensor on pin 3.)



3b

External digital sensors:

- can be used with any PICmicro chips (e.g. PIC16F84 or PIC16F627).
- are plugged in to the digital sensor port (lower white connector on the right-hand side of the PICmicro development board).
- requires corresponding part of jumper 14 set to digital (DIG);
  (The diagram shows all four inputs set for digital sensors.)
- In addition, jumper 1 must be removed.



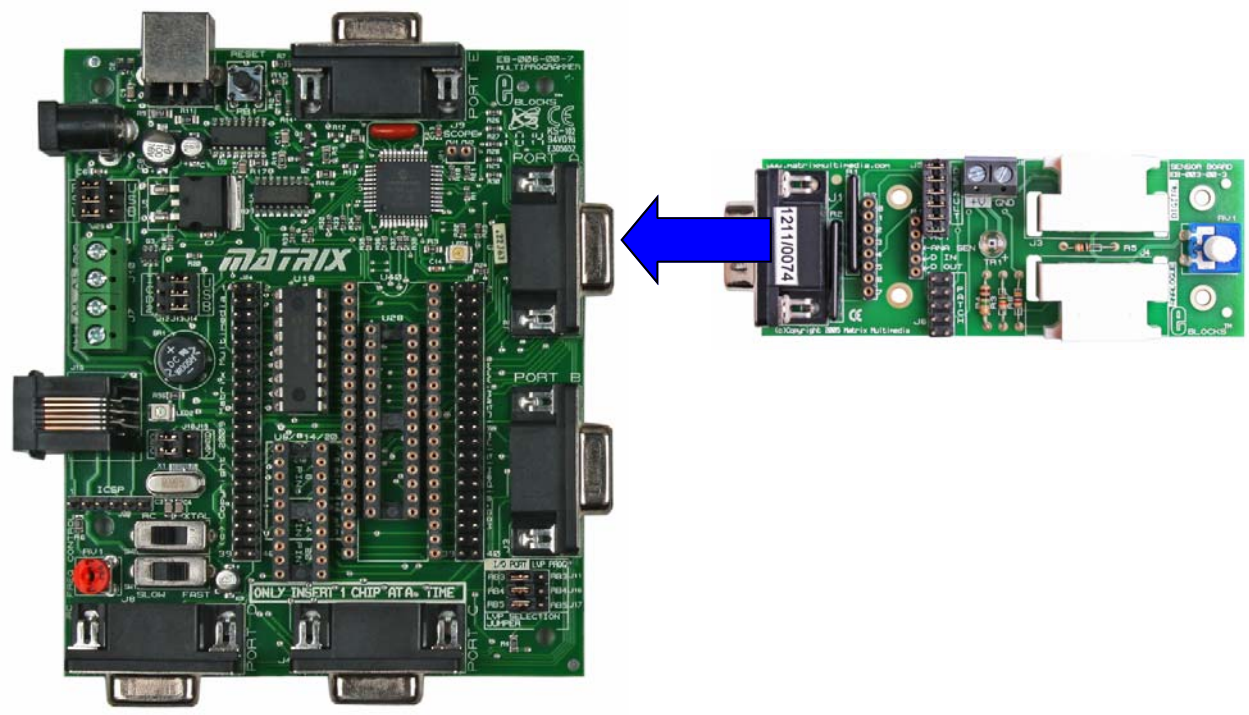
3c

The hardware

Sensors and Actuators

Using E-blocks

- The Actuators panel contains a D-type connector which can be connected directly to Up-stream boards like the EB006 if you are using an 8 bit PIC, or EB064 is you are using a dsPIC or other 16 bit device.
- To use the sensors described in this document you will need to connect a sensors board to the programmer board. You will need to consult the datasheet on the board you are using to find out which microcontroller port the sensor inputs are on. On most boards it is port A as shown below.



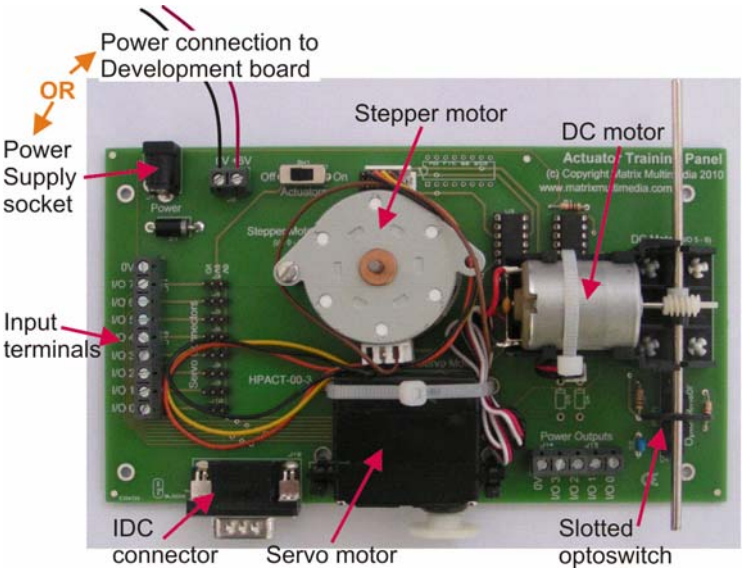
The hardware

Sensors and Actuators

Using the Actuator Training Panel

This panel has three purposes:

- it contains three different motors to allow students to investigate how various motor systems work;
- it provides an interface between the 'Alex' animated head or the 'TecArm' robot systems and the PICmicro development board;
- it provides power outputs for general purpose work such as powering external motors and relays.



4a

The on/off switch

- Setting the switch to the right 'On 'position turns power onto the actuators.
- Setting the switch to the left 'Off' position turns the actuators off and allows the Actuators Panel to be used with either the external power outputs or the Alex animated head /TecArm connection.
- Note that the power outputs and the servo connections for Alex / TecArm are always on.

How to connect the board

The 0V terminal and the eight Input/Output terminals on the left hand side of the Actuators Panel mirror the set of terminals found on the right hand side of the PICmicro development board. They can be connected together with ribbon cable, or with nine separate wire links. The table lists the Actuators Panel connections.

Actuator terminal	Description
0V	0V (ground) connection
I/O 0	Stepper motor - yellow
I/O 1	Stepper motor - brown
I/O 2	Stepper motor - black
I/O 3	Stepper motor - orange
I/O 4	Servo motor
I/O 5	DC motor - clockwise
I/O 6	DC motor - anticlockwise
I/O 7	Optosensor signal (output)

You can connect some or all of the terminals depending on what elements of the Actuators Panel you wish to use.

Alternatively, you can use an E-Blocks IDC cable (EB634) to connect the Actuators Panel to port C or D of a PIC chip such as the 16F877.

The hardware

Sensors and Actuators

Power outputs:

A set of four extra power output connections, and a 0V connection, can be found on the bottom edge of the Actuators Panel. These copy I/O connections 0-3, buffered by an L293D driver chip. They cannot be used as inputs.

Note that these are wired in parallel with the stepper motor so if you are using the power outputs for an application and you have the Actuators Panel switch in the ‘On’ position, you should expect the stepper motor to judder.

Power supply:

The actuators board requires a nominal 6V.

Using the external supply, the power connector is ‘positive outer’ – please make sure you have it the right way round before powering it up.

Testing the hardware

Sensors and Actuators

Actuators Panel test routine:

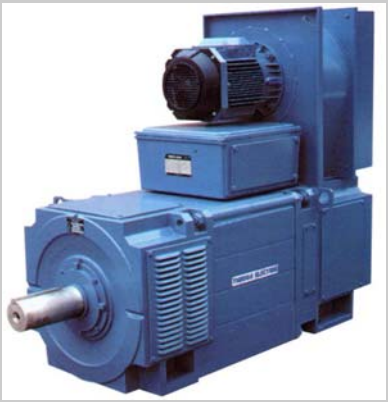
The routines ‘ActuatorsTest1.fcf’ (for Flowcode) and ‘ActuatorsTest1.asm’, supplied with the panel, run the following tests:

- Stepper motor clockwise and then counter-clockwise;
- DC motor clockwise and then counter-clockwise;
- Opto sensor (4 rotations of the DC motor);
- Servo motor 0º and then 90º.

Please use one of these to check that the Actuators Panel is fully operational.

Worksheet 1
Driving the DC motor

Sensors and Actuators



11a

DC motors use internal magnets and brushes to rotate the shaft, either clockwise or counter-clockwise. The motor turns continuously when a voltage is present.

Unlike stepper motors and servos, accurate positioning is very difficult to achieve. DC motors are normally use a sensor, to monitor position. One example is the car park barrier which trips a sensor when the raised arm is high enough.

The position of the motor on the Actuators Panel is controlled by an opto-electronic sensor, a disc with holes at known intervals, attached to a shaft driven by the motor. By counting pulses from the sensor, caused when holes pass between it and the light source, it is possible to calculate how far the shaft has rotated.

To control speed, the motor can be turned on and off in bursts. The longer the motor is off compared on, the slower the motor rotates. This is called Pulse Width Modulation (PWM).

If the motor is rotating too quickly, it may overshoot the intended position before it stops. PWM is a useful technique here. The motor is slowed as it nears the stop point, increasing positional accuracy. (This is the basis for fuzzy logic.)

Over to you:

- Connect the Actuators Panel to Port C of the Development Board with an IDC cable.
- A logic1 signal on I/O5 makes the motor rotate clockwise.
- A logic1 signal on I/O6 turns the motor counter-clockwise.
- A signal on both I/O5 and I/O6 is ignored and there is no movement.
- The motor will run at full speed as long as the logic 1 signal is maintained.
- To test this, create a Flowcode program to make the motor rotate first clockwise and then counter-clockwise. To do this:
    - use the crystal clock on the Development Board;
    - add an infinite loop, and inside it :
        - output value 1 to Port C, bit 5;
        - insert a delay for 5s;
        - output value 1 to Port C, bit 6;
        - insert another delay for 5s.

Worksheet 1
Driving the DC motor

Sensors and Actuators

Over to you again:

Using PWM:

- the motor speed depends on the mark-to-space ratio.
- For example, if the signal is ‘on’ for 1ms and then ‘off’ for 9ms, then the motor is on for only one tenth of the time and so runs at about 1/10th of full speed.

The opto-sensor:

- is connected to I/O7.
- registers ‘high’ (logic 1) with an uninterrupted beam, and ‘low’ (logic 0) when the beam is blocked.
- The shaft-mounted disc has holes at 90-degree intervals.
- When the hole passes the sensor the signal changes from low to high.

The tasks:

- Modify the program so that the direction of rotation is controlled by pressing one of two switches - one for clockwise rotation, one for counter-clockwise rotation. (Level 2)
- Add a refinement where two further switches make the motor go forwards and backwards at one quarter speed using Pulse Width Modulation. (Level 2)
- Use feedback from the opto-sensor on Port B bit 7 to make the motor rotate through two complete clockwise revolutions when a switch on port A is pressed.
- Refine this program by making the motor slow down as it is about to reach its target. Monitor how quickly, and how accurately you get to two complete revolutions.

For your records:

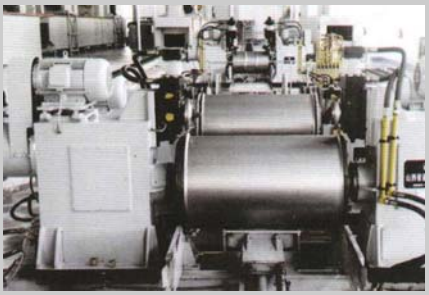
Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high-level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Worksheet 2
Driving the stepper motor

Sensors and Actuators

Stepper motors rotate in small steps, as the name suggests, making possible accurate control of position and rotational speed.
The picture shows them used in an industrial production line.



13a

They:

- use a set of internal electro-magnets to attract the rotor;
- activate these in a specific order to move the rotor to the next position;
- are designed to step a set number of degrees in a single step.

Repeating the sequence of signals causes the rotor to move round in a circle.

The stepper motor on the Actuators Panel moves in 15° steps and does not support half steps. To rotate it by one step (15°) clockwise, use the following sequence:

	Orange (I/O 3)	Yellow (I/O 0)	Brown (I/O 1)	Black (I/O 2)	Input on pins 0 to 3 (bin, hex, dec)
Step 1	1	0	1	0	1010_2, $0A_{16}$, 10_{10}
Step 2	0	1	1	0	0011_2, 03_{16}, 3_{10}
Step 3	0	1	0	1	0101_2, 05_{16}, 5_{10}
Step 4	1	0	0	1	1100_2, $0C_{16}$, 12_{10}

To rotate the motor counter clockwise, simply reverse the sequence.
To move the stepper through 'x' steps, repeat the above sequence 'x' times.

Over to you:

- Connect the Actuators Panel to Port C of the Development Board with an IDC cable.
- Create a Flowcode program to make the stepper motor rotate clockwise. To do this:
    - use the crystal clock on the Development Board;
    - add an infinite loop, and inside it :
        - output value 10 to Port C;
        - add a 100ms delay;
        - output value 3 to Port C;
        - add another 100ms delay;
        - output value 5 to Port C;
        - add a third 100ms delay;
        - output value 12 to Port C;
        - add a fourth 100ms delay.

This flowchart is shown in the diagram on the next page.

The direction of rotation is reversed by reversing the order of the values outputted to Port C.

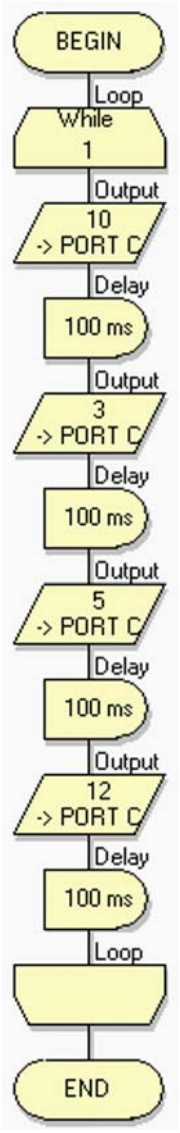
Worksheet 2

Driving the stepper motor

Sensors and Actuators

Over to you again:

The diagram shows the program described on the previous page:



Modify the original program to:

- use a switch to turn the motor clockwise, step-by-step, when it is pressed. (Level 2)
- use a second switch to turn the motor counter-clockwise, step-by-step, when it is pressed. (Level 2)

Write a new program to:

- use one switch to turn the motor forwards by one 15-degree step, and a second switch to turn it backwards by one 15-degree step, when pressed. (Level 2)

14a

For your records:

Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Worksheet 3
Driving the servo motor

Sensors and Actuators

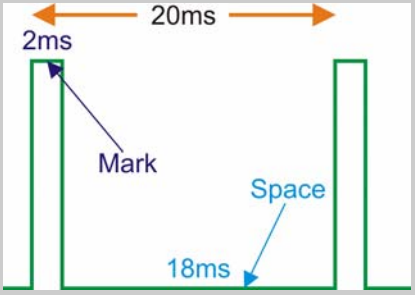

15a

Servo motors consist of a motor, driving an arm, or disc, attached to it. Unlike DC motors, the servo motor does not rotate freely but instead turns through an angle dependant on the mark-to-space ratio of the signal sent to it. Attached mechanical linkage can then convert this rotation into linear movement.

The servo motor on the Actuator Panel is driven from I/O line 4. It requires a square pulse of twenty milliseconds duration.

The 'mark', the high voltage section at the beginning of the 20ms pulse, determines the angle the servo moves through:

- a signal length of 1ms moves the servo arm 0°
- a signal length of 1.5ms moves the arm 90°
- a signal length of 2ms moves the arm 180°



15b

The 'space', the second section of the square pulse, is a low voltage signal that lasts for the remainder of the 20ms pulse, (normally between 18 and 19 ms).

The diagram shows a pulse with a mark-to-space ratio of 1:9, giving a rotation of 180^0.
To maintain rotor position, the whole pulse needs repeating every 20ms.

Servo motors normally have internal stops to prevent movement outside of the 180° arc.

Over to you:

- Connect the Actuators Panel to Port C of the Development Board with an IDC cable.
- Create a Flowcode flowchart to make the servo rotate through 90^0 and stay there.
  To do this, add an infinite loop, and inside it :
    - add a second loop with a loop count of 200, and inside it:
        - output value 1 to Port C, bit 4;
        - insert two delays, a 1ms, and a 500μs, to give a total delay of 1.5ms;
        - output value 0 to Port C, bit 4;
        - insert two delays, a 18ms, and a 500μs, to give a total delay of 18.5ms;
    - add a third loop, again with a loop count of 200, and inside it:
        - output value 1 to Port C, bit 4;
        - insert a delay for 1ms;
        - output value 0 to Port C, bit 4;
        - insert a delay for 19ms;

This flowchart is shown on the next page.

Worksheet 3

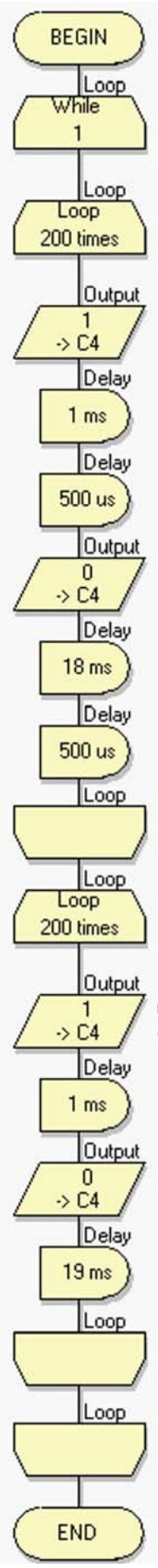
Driving the servo motor

Sensors and Actuators

Over to you again:

The diagram shows the program described on the previous page:

Modify the original program to use four switches to select four different mark-to-space ratios and hence select four different angles of rotation (Level 2)



16a

For your records:

Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Worksheet 4
Measuring heart-rate

Sensors and Actuators

The Heart-rate Monitor (product code HSEHR) is ideal for determining the heart-rate even when the volunteer is exercising.

17a

It consists of a wireless transmitter belt and a receiver module. The transmitter belt senses electrical signals generated by the heart, much like an ECG. A signal is transmitted every time a heartbeat is detected. The receiver has a range of 80 to100cm.

Over to you:

- Read the information supplied with the Heart-rate Monitor, and with the PICmicro Development Board. Make sure you understand how the heart rate sensor is connected to the Development Board.
- The heart-rate monitor is an analogue sensor and so requires a PICmicro device with A/D inputs. Make sure you have an appropriate device in the development board.
- Devise a Flowcode program that reads an analogue value and flashes an LED on port B every time a big enough signal is detected. To do this
  - use the crystal clock on the Development Board;
  - configure the Development Board for analogue sensors, as described on page 3.
  - add an ADC input component and a LED;
  - set up an infinite loop;
  - use one of the ADC component macros to read the value from the Heart-rate monitor into a suitable variable;
  - test whether this value exceeds a threshold value (e.g. 100);
  - light the LED if it does, and turn it off if it doesn’t;
  - incorporate a suitable delay to allow you to observe what is happening.
- Use the on-board potentiometer RV3 to test your program before connecting the Heart-rate Monitor. (The signal from the on-board potentiometer appears on pin AN1, and so the program needs to read from that input.).
- Place the chest band/transmitter on the volunteer. For maximum efficiency, wet the two electrodes with a 5% saline solution.
- Plug the receiver into the analogue input of the PICmicro Development Board.
- Connect an oscilloscope to RA4 and view the analogue signal from the Heart-rate Monitor.
- Modify the program to read from this input.
- Adjust the threshold value in your program if the LED flashing seems too erratic or if it is not flashing at all.

Worksheet 4
Measuring heart-rate

Sensors and Actuators

Over to you again:

- Write a simple program, in Flowcode, C or Assembler, that displays the sensor value on the LCD display. (In Flowcode, you need to add a LCD to the Flowcode panel, and use the ‘Start’, ‘Clear’ and ‘PrintNumber’ macros.) (Level 2)
- Using interrupts or another timing mechanism, create a program in Flowcode, C or Assembler, to calculate the time between pulses. Display this value on the LCD display. Then manipulate this value to provide a reading for the number of beats per minute. (Level 2)
- Average the ‘beats per minute’ value over a number of beats (say for 5 to 10 beats,) to give a steadier and more accurate value of heart-rate. (Level 3)
- Once a valid display of beats per minute is achieved, create a program in Flowcode, C or Assembler, that monitors heart-rate and lights an LED if it is too fast or too slow. (Level 3)

For your records:

Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Worksheet 5
Measuring temperature

Sensors and Actuators

The temperature probe (product code HSTMP) has a sealed stainless steel shaft and tip that can be used in organic liquids, salt solutions, acids, and bases.

19a

It is designed to measure temperatures from –25 to 125ºC, with a maximum recommended temperature of 150ºC.

Over to you:

- Read the information supplied to understand how the temperature probe is connected to the Development Board.
- This is another analogue sensor and so requires a PIC device with A/D inputs. Make sure you have an appropriate device in the development board.
- Devise a Flowcode program that monitors the temperature of a bath and lets you know if it is too cold or too hot. To do this
    - choose an optimum temperature for a bath (say 50°C);
    - heat a beaker of water to this temperature, monitoring it with a thermometer;
    - use the crystal clock on the Development Board;
    - configure the Development Board for analogue sensors, as described on page 3.
    - add an ADC input component;
    - set up an infinite loop;
    - use an ADC component macro to read the temperature value into a suitable variable;
    - use your program to find out the value of the temperature probe signal when it has been in the beaker of hot water for a few minutes;
    - choose a ‘too hot’ temperature and a ’too cold’ temperature, for example 55°C and 45°C respectively;
    - use your program to find the temperature probe values at these temperatures, as before;
    - test the value of the signal against these two threshold values;
    - light one LED if it is too hot, a different one if it is too cold, and a third when it is neither of these;
- Use the on-board potentiometer RV3 to test your program.
- Plug in the temperature probe. Its input is received as an analogue signal on pin A4.
- Connect an oscilloscope or a multimeter to pin A4. The DC signal should vary as you warm up the sensor in your hand.
- Use hot and cold water to create a "bath" at around 50°C in a beaker, using your program to indicate temperature. Check the result with a thermometer.

Worksheet 5
Measuring temperature

Sensors and Actuators

Over to you again:

- Write a simple program, in Flowcode, C or Assembler, that displays the value from the temperature probe onto the LCD display. Update this value regularly (e.g. every second). Note that this is an uncalibrated value and does not represent the temperature in °C or °F. (In Flowcode, you need to add a LCD to the Flowcode panel, and use the 'Start', 'Clear' and 'PrintNumber' macros.) (Level 2)
- Construct a temperature chart for 0°C to 100°C using the data on the temperature probe or by calibrating the probe with a known thermometer. (Level 2)
- Place the probe in a cup of boiling water and remove the heat from the beaker so it begins to cool. Note down the reading of the temperature probe at regular intervals (e.g. every 30 seconds for about 20 minutes). Convert each of these values to °C using your chart and plot the a graph of your results. Repeat this experiment to see if you can speed up the cooling process by blowing on the cooling water or by placing next to an open window. (Level 2)
- There are lots of applications for monitoring temperature once you have created a calibration chart. Develop the temperature display program to display the actual temperature in °C. The best way to do this is to form a look-up table (see Using Macros page for example code for creating a computed-GOTO jump table). (Level 3)

For your records:

Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Worksheet 6

Timing using the photo-gate

Sensors and Actuators

Photo-gates can be used to study free fall, rolling objects, air track collisions, pendulums, etc.

The photo-gate sensor (product code HSVPG) consists of the sensor unit, which sends an infra-red beam between the two arms, and mounting rod .

21a

Photo-gates become particularly useful when used with the smart pulley adaptor (code HSSPA) for measuring speed, the picket fence (code HSPF) for measuring ‘g’ or the bar tape (code HSTAPE) which replaces ticker tape in speed, and acceleration investigations. The ‘smart’ pulley attaches to the photo-gate, and has a slotted wheel, which passes through the photo-gate’s beam.

Over to you:

- Read the information supplied to understand how the photo-gate is connected to the Development Board.
- This is a digital sensor, and the Development Board must be configured accordingly, as described on page 3. When the infra-red beam is received, the sensor output is high, i.e. pin A4 = logic 1. When the beam is blocked, the output is low and pin A4 = logic 0.
- With the photogate plugged in to the Development Board, use an oscilloscope or a multimeter to monitor input pin A4. The DC signal should vary when you block the infrared beam.
- The photogate acts like a simple switch attached to pin A4 of the PIC. When the beam is broken, the value of A4 is low (logic 0). When it is not broken, A4 is high (logic 1). Devise a Flowcode program that mirrors the status of the sensor signal onto output pin B4.
  To do this
  - use the crystal clock on the Development Board;
  - set up an infinite loop;
  - read the logic level present on pin A4 to a suitable variable;
  - output this variable to a LED connected to Port B, pin 4;
- Modify the program to create a simple burglar alarm. When the signal is broken, all LEDs on Port B light and stay on.
- Incorporate a ‘Reset’ switch (on Port A bit 0), which turns all LEDs off.

Worksheet 6
Timing using the photo-gate

Sensors and Actuators

Over to you again:

- Write a program to count the number of pulses detected on input pin A4 when the 'smart' pulley is attached to the photo-gate. (Level 2)
  - Display this value on the LCD display .
  - Make it reset to zero each time a switch on the Development Board is pressed.
- Modify the program to count the number of revolutions of the pulley wheel. (Level 2)
  - Calculate the circumference of the pulley wheel.
  - Develop a program that indicates the approximate distance travelled when the smart pulley and photo-gate are run along the desk.
- Enhance the burglar alarm program to provide flashing lights and an audio output when triggered. (Level 2)
  - Add a countdown before the alarm is activated.
  - Add a 'PIN' password, entered on Development Board switches, to disable the alarm.
- The pulley wheel concept can be extended to measure rotational speed. (Level 3) Develop a program that:
  - calculates the time interval between the pulses from the pulley, on bit A4 using interrupts;
  - converts this into revolutions per second, and display the result.

For your records:

Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Worksheet 7
Measuring motion

Sensors and Actuators

The Motion Detector (product code HSMD) functions like the automatic range finder on a camera.

It emits ultrasonic pulses and waits for an echo. The time it takes for the reflected pulses to return is used to calculate distance, velocity, and acceleration.

The device has a range of 0.4 to 6.0 metres, within a detection cone of 15 to 20º.

23a

Over to you:

- Read the information supplied to understand how the Motion Detector is connected to the Development Board.
- Again, this is a digital sensor, and the Development Board must be configured accordingly, as described on page 3.
- With the Motion Detector plugged in to the Development Board, use an oscilloscope to monitor input pin A4. If you having trouble receiving the echo, check that there is no object in the detection cone that may be interfering with the experiment.
- Write a Flowcode program that displays the time in milliseconds for the echo to return
  To do this:
  - use the crystal clock on the Development Board;
  - set up an infinite loop;
  - wait for a switch, connected to Port A bit 2, to be pressed. This causes the transmission of an ultrasound pulse.
  - check to see if the echo has been received on Port A bit 4;
    - if it has not:
      - increment a suitable variable;
      - wait for 1ms;
      - return and check for an echo.
    - if it has:
      - display the value of the variable (roughly, the time taken in ms,) on the LCD screen.

Worksheet 7
Measuring motion

Sensors and Actuators

Over to you again:

- Modify the program to display the distance to the object in centimetres. Assume that ultrasound travels at 340m/s, or 34.0 cm/ms, in air. (Level 2)
- Modify the original program to sound an alarm when something is too close to the sensor. (Level 2)
- Write a program to display the speed of a slowly moving object. To do this, sample the distance travelled repeatedly, at intervals of 10ms, say. (Level 3)

For your records:

Presenting your work:

- In developing your work, you should keep copies of all printouts of code and flowcharts.
- If you are using a low level language (Assembly or C) then include high level system diagrams showing the algorithms used.
- All code should be properly commented.
- For each task undertaken, keep a written record of problems encountered and their solutions.

Tutor's notes

Sensors and Actuators

About this course

Introduction

This workbook is intended to reinforce the learning that takes place in the classroom or lecture room. It provides a series of practical activities and investigations that use the PIC Development Board (HP488) or E-blocks and the Actuators Panel (HPACT).

Aim

The workbook aims to introduce students to the principles of sensors and actuators, and how these can be controlled by PIC microcontrollers.

Prior Knowledge

Students are expected to have:

Some previous experience in programming in Flowcode.

- an understanding of analogue and digital quantities;
- familiarity with features of the PIC Development Board.

Learning Objectives

On successful completion of this course the student will be able to:

- write a program to make a DC motor rotate clockwise or counter-clockwise;
- explain pulse-width modulation (PWM);
- control the speed of a DC motor using PWM;
- use feedback from an opto-sensor to control the position of a DC motor;
- describe the operating principles of the stepper motor;
- write a program to make a stepper motor rotate clockwise or counter-clockwise through a given number of steps;
- calculate the mark-to-space ratio for a given pulse;
- describe the operation of the servo motor, and the significance of mark-to-space ratio;
- write a program to make a servo motor rotate through a given angle;
- set up the Heart-rate Monitor and connect it to the Development Board;
- write a program to monitor and display pulses from a volunteer's heart,
- write a program to display heart-rate in beats per minute, and set alarms for high and low heart-rate;
- carry out simple signal processing, turning a varying analogue quantity into a digital one;
- write a program to monitor the temperature of a liquid, and set alarms for high and low temperature;
- use look up tables to manipulate data;
- use digital signals from a sensor to control the behaviour of a microcontroller;
- write programs using photo-gates and motion detectors to display distance travelled and speed.

Tutor's notes

Sensors and Actuators

Approach:

The worksheets in this document allow students to carry out a number of exercises using sensors and actuators. These can be used as a framework of assessment in many electronics, control, and robotics courses. They require that students can reference the information in this document and in the PIC Development Board datasheet, which is included on the floppy disk/CD, supplied with the board.

We do not supply completed code, or flowcharts, for the sensors although code has been written to verify that these exercises are practical.

We hope that you will find the exercises involving the DC motor and feedback loop particularly effective in teaching the use of real feedback systems.
Please do contact us with feedback and comments.

Sensors overview:
The four sensors have been chosen because they require students to gather real world data using different programming strategies:

Sensor	Output	Coding strategy
Temperature probe	Simple voltage divider	A/D conversion, calibration, look up table, display
Motion detector	Digital pulse correlating to distance	Pulse time measurement, conversion, display
Heart rate monitor	Analogue voltage pulse	Data slicing, timing,
Photogate	Digital pulse dependant on IR or not	Various

These sensors are for educational use only.

Tutor's notes

Actuators overview:

The use of the simple motors on the actuators panel is dealt with here.

The actuators on the panel are:

Actuator	Output/action	Coding strategy
Stepper motor	Full step	Full step, Forward and reverse,
Servo motor	Angle control	Various
DC motor	Various	Forward and reverse, Speed control using PWM, Feedback control

Using the worksheets

These worksheets are fully editable – please feel free to modify them for your own use.

They are split into different levels of accomplishment. The amount of work that your students will get through will depend on their competence with programming, and how long you want them to spend on the task.

The worksheets can be addressed in Assembly, C or Flowcode.

The mix of problems included are usable over a wide range of courses from Schools Design and Technology through to University microprocessor courses.

The worksheets are in no particular order. Please read them and judge for yourself which your students should start with.

Students will need the information on the sensors and actuators given in previous sections as well as the worksheet itself. You may choose to give them pre-written routines or macros to help them to complete the worksheets.

It is expected that the worksheets are printed / photocopied, preferably in colour. Students should retain their own copy of the entire workbook.

Tutor's notes

Sensors and Actuators

What students will need:

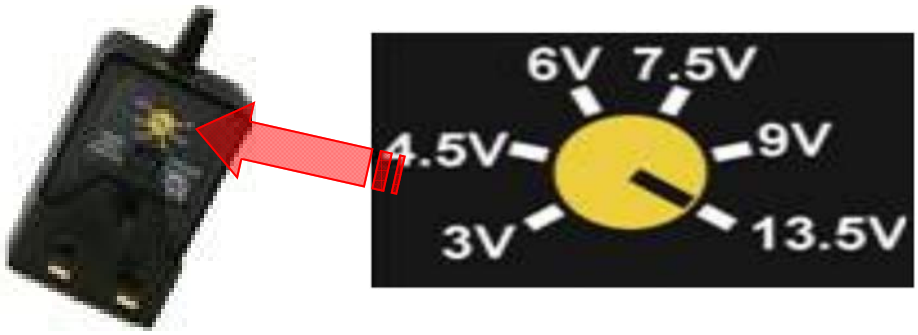
Students will need:

- HP 488 PIC Development Board, or E-blocks equivalent
- HPACT Actuators Training Panel;
- HP5328 Adjustable DC power supply;
- PIC 16F877 Microcontroller, or equivalent;
- EB634 IDC cable;
- HSEHR Heart-rate monitor;
- HSTMP Stainless steel temperature probe;
- HSVPG Fast response photo-gate;
- HSMD Motion detector.
- LK1110 Multimeter or equivalent;
- LK6730 Pico 4000 virtual oscilloscope, or equivalent.

Power sources:

The HP5328 adjustable DC power supply is recommended for use with the activities described in this workbook. Its output voltage can be adjusted to provide outputs of either 3 V, 4.5 V, 6 V, 7.5 V, 9 V or 13.5 V, with currents typically up to 1 A. The voltage is changed by turning the selector dial just above the earth pin until the arrow points to the required voltage.

In the diagram below, the DC power supply has been adjusted to provide an output of 13.5 V DC.



28a

28b

Tutor's notes

Sensors and Actuators

Worksheet	Notes for the Tutor	Timing
1	**Equipment required** Development Board, standard PIC device e.g. PIC16F84, Actuators Panel. **Assumed knowledge** Reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of PIC and Development Board architecture. Mark-to-space ratio. **Objectives** To teach students how to control a DC motor and incorporate it into real systems.	**60 - 90 minutes**
2	**Equipment required** Development Board, standard PIC device e.g. PIC16F84. Actuators Panel. **Assumed knowledge** Reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of PIC and Development Board architecture. Binary numbering. **Objectives** To teach students how to control a stepper motor and incorporate it into real systems.	**60 - 90 minutes**
3	**Equipment required** Development Board, standard PIC device e.g. PIC16F84, Actuators Panel. **Assumed knowledge** Reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of PIC and Development Board architecture. Mark-to-space ratio. **Objectives** To teach students how to control a servo motor and incorporate it into real systems.	**60 - 90 minutes**

Tutor's notes

Sensors and Actuators

Worksheet	Notes for the Tutor	Timing
4	**Note**: Some institutions with mixed sex classes have reported embarrassment/reluctance in using the Heart-rate Monitor, as it needs to be placed under student's clothing. Please consider this issue when using the heart rate monitor. **Equipment required** Development Board, Heart-rate Monitor, PIC device with built-in A/D converters e.g. PIC16F874 or PIC16F877. **Assumed knowledge** Reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of how to convert analogue signals into digital. Understanding of PIC and Development Board architecture. Binary and hexadecimal number systems. **Objectives** To understand the process of converting a time-varying analogue quantity into a digital one. To carry out simple signal processing to turn a varying analogue quantity into a digital one. To manipulate this data to provide meaningful readings for human beings.	**45 - 60 minutes**
5	**Equipment required** Development Board, Temperature probe, PIC device with built-in A/D converters e.g. PIC16F874 or PIC16F877. Freezer spray (optional). **Assumed knowledge** Reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of how to convert analogue signals into digital. Understanding of PIC and Development Board architecture. Binary and hexadecimal. **Objectives** To understand the process of converting a simple analogue quantity into a digital one To develop a 'control system' based on this data To use look up tables to manipulate this data to provide meaningful readings for human beings	**45 - 60 minutes**

Tutor's notes

Sensors and Actuators

Worksheet	Notes for the Tutor	Timing
6	**Equipment required** Development Board, Photo-gate sensor and smart pulley, standard PIC device e.g. PIC16F84, spring, string and weights. **Assumed knowledge** A reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of PIC and Development Board architecture. **Objectives** To understand the process of detecting a digital signal. To monitor and record time-varying digital signals. To convert time-varying digital signals into meaningful data for human beings. To carry our more complex processing on irregular digital signals.	**60 - 90 minutes**
7	**Equipment required** Development Board, Motion detector, standard PIC device e.g. PIC16F84, a football and a tape measure. **Assumed knowledge** A reasonable level of competence in coding for a PIC microcontroller in the language of choice – Assembler, C, or Flowcode. Understanding of PIC and Development Board architecture. **Objectives** To understand and use an active sensor in a simple system. To develop real systems based on active sensors.	**60 - 90 minutes**