**NZQA**

**Approved**

Achievement standard: 91078 Version 3

Standard title: Implement basic interfacing procedures in a specified electronic environment

Level: 1

Credits: 3

Resource title: Cooling down the car

Resource reference: Digital Technologies VP-1.48 v2

Vocational pathway: Manufacturing and Technology

|  |  |
| --- | --- |
| Date version published | February 2015 Version 2  To support internal assessment from 2015 |
| Quality assurance status | These materials have been quality assured by NZQA.  NZQA Approved number A-A-02-2015-91078-02-7347 |
| Authenticity of evidence | Assessors/educators must manage authenticity for any assessment from a public source, because learners may have access to the assessment schedule or exemplar material.  Using this assessment resource without modification may mean that learners’ work is not authentic. Assessors/ educators may need to change figures, measurements or data sources or set a different context or topic to be investigated or a different text to read or perform. |

Vocational Pathway Assessment Resource

Achievement standard: 91078

Standard title: Implement basic interfacing procedures in a specified electronic environment

Level: 1

Credits: 3

Resource title: Cooling down the car

Resource reference: Digital Technologies VP-1.48 v2

Vocational pathway: Manufacturing and Technology

Learner instructions

# Introduction

This assessment activity requires you to implement basic interfacing procedures to make a sensor-controlled cooling fan system for a car.

You are going to be assessed on how efficiently you implement basic interfacing procedures to make a sensor-controlled cooling fan system for a car.

The following instructions provide you with a way to structure your work to demonstrate what you have learnt to allow you to achieve success in this standard.

Assessor/educator note: It is expected that the assessor/educator will read the learner instructions and modify them if necessary to suit their learners.

# Task

You will use basic interfacing procedures to make a sensor-controlled cooling fan system for a car using basic electronic components and a microcontroller that performs to the specifications.

Basic ‘interfacing procedures’ refer to the selection, testing and debugging of the hardware and software that allows different devices to work together to meet the given specifications for the sensor-controlled car cooling system.

You will present a portfolio which will include evidence of how you use datasheets or calculations to help choose appropriate components (types and values) for the interface. Include in your portfolio evidence of the writing of your well-structured, functional and clearly annotated interface software.

## Specifications

Specifications must be agreed with the assessor/educator prior to the implementation of the interfacing procedures.

Specifications for the sensor-controlled cooling fan system include:

* a system that indicates when the engine’s water temperature reaches approximately 90 degrees Celsius. (The temperature at which the cooling fan should switch on)
* sensors suitable for detecting the temperature of the engine
* a system for starting the fan when the water temperature reaches 90 degrees Celsius
* a remote indicator, using a LED, that shows when the fan is operating
* a short warning buzzer that buzzes for 1 second to warn the driver that the fan is about to switch on
* a circuit that provides a switch so the driver can manually operate the cooling fan
* a switch for manually starting the fan.

## Steps for implementing and testing

Choose appropriate component types and values for the sensor-controlled cooling fan system and test and debug a functional model.

Use data sheets or calculations to assist in choosing appropriate component types and values for the sensor-controlled cooling fan system.

Write well-structured, clearly annotated, and readily understandable software.

Select appropriate electronics components for an interface that links the sensor-controlled cooling fan system to the microcontroller. Show how you have used data sheets or calculations to assist in selecting these components. You may use circuit diagrams to assist in showing this.

Write a computer program that controls the sensor-controlled cooling fan system and allows it to work according to the specifications provided. The software interface needs to be well structured, readily understandable, and clearly annotated. This means your computer program should:

* be clearly set out and correctly indented
* include comments that explain exactly what the program is doing at each step
* be labelled so that it is easy to read.

Test and debug a working model of the sensor-controlled cooling fan system. This means checking that the sensor-controlled cooling fan system works according to the specifications and taking corrective action where it is not performing as expected.

## Performing and completing the task

As you perform the task, make notes and gather evidence for inclusion in your portfolio.

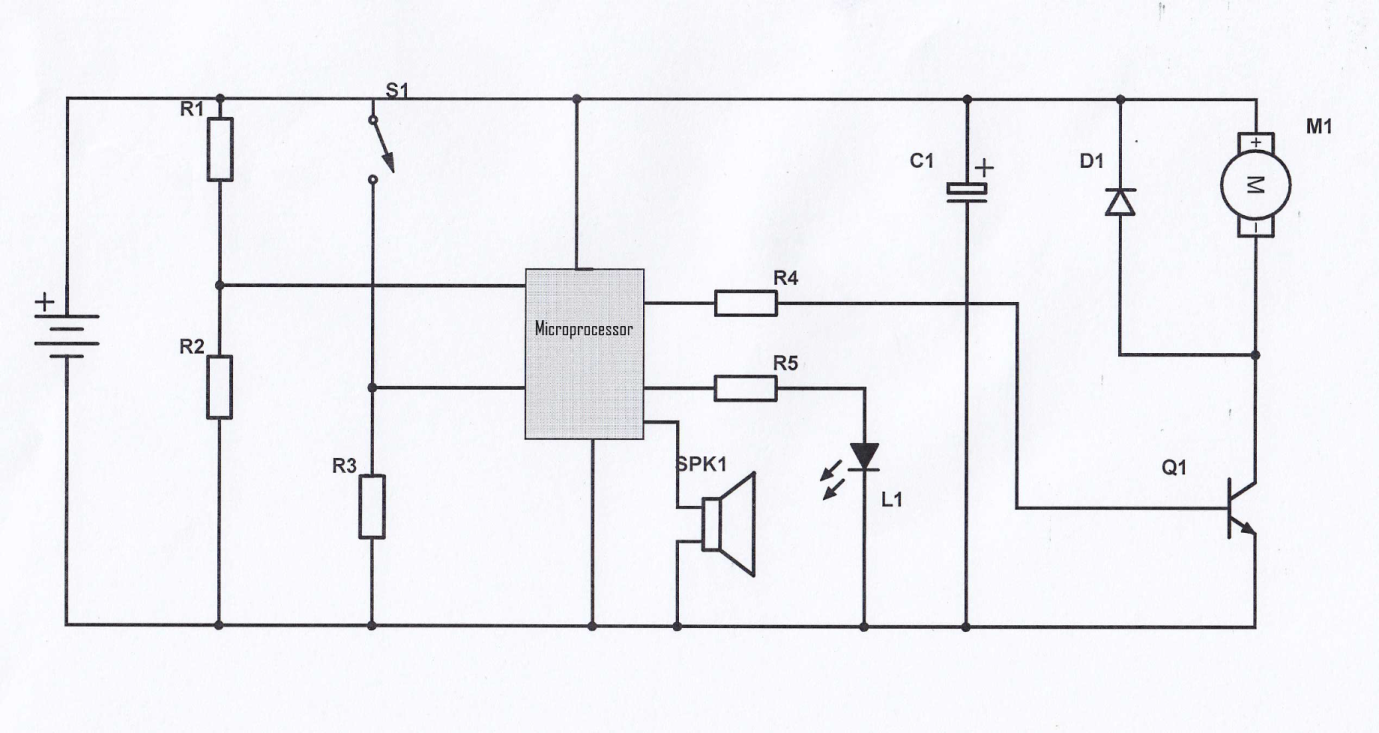
## Presentation

Present your evidence as a portfolio, which may include:

* annotated diagrams and photographs
* short video clips
* journal entries that describe testing and debugging activities
* your responses to assessor/educator questions
* interview notes from any scheduled assessor/educator consultation
* code for your computer program.

# Resources

Use the circuit below, if required.



## Useful websites

<http://www.arduino.cc/en/>

[www.electronics-tutorials.com/basics/basic-electronics.htm](http://www.electronics-tutorials.com/basics/basic-electronics.htm)

<http://williamson-labs.com/>

[www.electronicsclub.info](http://www.electronicsclub.info)

[www.technologystudent.com](http://www.technologystudent.com)

Vocational Pathway Assessment Resource

Achievement standard: 91078

Standard title: Implement basic interfacing procedures in a specified electronic environment

Level: 1

Credits: 3

Resource title: Cooling down the car

Resource reference: Digital Technologies VP-1.48 v2

Vocational pathway: Manufacturing and Technology

Assessor/Educator guidelines

# Introduction

The following guidelines are supplied to enable assessors/educators to carry out valid and consistent assessment using this internal assessment resource.

As with all assessment resources, education providers will need to follow their own quality control processes. Assessors/educators must manage authenticity for any assessment from a public source, because learners may have access to the assessment schedule or exemplar material. Using this assessment resource without modification may mean that learners' work is not authentic. The assessor/educator may need to change figures, measurements or data sources or set a different context or topic. Assessors/educators need to consider the local context in which learning is taking place and its relevance for learners.

Assessors/educators need to be very familiar with the outcome being assessed by the achievement standard. The achievement criteria and the explanatory notes contain information, definitions, and requirements that are crucial when interpreting the standard and assessing learners against it.

# Context/setting

This activity requires learners to efficiently implement basic interfacing procedures to make a working sensor-controlled cooling fan system using basic electronic components and a microcontroller. The sensor-controlled cooling fan system must perform to the specifications outlined in the task.

# Conditions

None.

# Resource requirements

Learners should be provided with:

* access to a computer
* a microcontroller system that includes a simple switch and a temperature sensor that can detect the heat of the water in the cooling system, and is able to control a relay or motor
* access to a digital camera and/or video camera to photograph portfolio evidence.

The list of resources may be adapted to meet the needs of your learners. Some learners may have cameras on their mobile phones. These can be used to document the process.

Electronic catalogues and supplies are available at:

[www.Surplustronics.co.nz](http://www.Surplustronics.co.nz)

[www.Activecomponents.com](http://www.Activecomponents.com)

[www.Mailtronics.co.nz](http://www.Mailtronics.co.nz)

[www.electroflash.org.nz/](http://www.electroflash.org.nz/)

Learners will also require:

* internet access
* materials to build their cooling system
* an electronic breadboard
* electronic circuit simulation software, for example:

[www.falstad.com/circuit/](http://www.falstad.com/circuit/)

[www.yenka.com](http://www.yenka.com)

* software for schematic capture, for example:

[www.cadsoft.de](http://www.cadsoft.de)

The following resources may also be useful:

* *An Introduction to Practical Electronics, Microcontrollers, and Software Design* by Bill Collis (available at [www.techideas.co.nz](http://www.techideas.co.nz)).
* *PICAXE Microcontroller Projects for the Evil Genius* by Ron Hackett (available at [www.amazon.com](http://www.amazon.com)).

# Additional information

The specifications need to be agreed prior to the implementation of interfacing procedures. They may be provided or developed in negotiation with the learner. When developing specifications, it is essential that:

* the specifications relate to the monitoring and control of variables in both hardware and software
* the electronic environment includes a functional combination of hardware and embedded software
* the specifications define the functional qualities required.

Learners are required to collect portfolio evidence as they complete the task. Guidance may be provided around what the evidence should look like.

## Other possible contexts for this vocational pathway

This context can be easily adapted for other electronic environments where there is a combination of hardware and embedded software that performs to specifications, for example:

* a building burglar alarm system
* a temperature controlled greenhouse
* a farm gate
* temperature control for an industrial spray painting workspace.

# Assessment schedule: Digital Technologies 91078 – Cooling down the car

|  |  |  |
| --- | --- | --- |
| Evidence/Judgements for Achievement | Evidence/Judgements for Achievement with Merit | Evidence/Judgements for Achievement with Excellence |
| The learner implements basic interfacing procedures in the sensor-controlled cooling fan system by:   * choosing appropriate component types and values for the interface   For example:  *I found that the Thermistor has a resistance of 68 kΩ when the temperature was at 100 degrees Celsius and 4 kΩ when the temperature was 0 degrees Celsius. So I chose a resistor in between these two values to use in the voltage divider. I used a 30 kΩ resistor.*   * writing basic functional interface software given simple program structures   For example:  See Appendix 1 for an example illustrating different versions of a code fragment.   * testing and debugging a functional model of the interface   For example:  *I found that when the sensor was placed on top of the radiator, the fan worked perfectly, but when I put it on top of the engine the fan ran all the time. I fixed this by making a little shield for the sensor so that it was protected from the heat of the engine bay. I also found that the fan worked OK one day and not the next. This was because the current draw of the fan motor was too great for the transistor, which was failing under load. I fixed this by adding a relay to control the fan motor. This made the fan work correctly when it was required to cool down the engine (see annotated photographs of this).*  *The above expected learner responses are indicative only and relate to just part of what is required*. | The learner skilfully implements basic interfacing procedures in the sensor-controlled cooling fan system by:   * choosing appropriate component types and values for the interface   For example:  *I found that the Thermistor has a resistance of 68 kΩ when the temperature was at 100 degrees Celsius and 4 kΩ when the temperature was 0 degrees Celsius. So I chose a resistor in between these two values to use in the voltage divider. I used a 30 kΩ resistor.*   * writing annotated, functional, and readily understandable interface software given simple programme structures   For example:  See Appendix 2 for an example illustrating different versions of a code fragment.   * testing and debugging a functional model of the interface for the sensor-controlled cooling fan   For example:  *I found that when the sensor was placed on top of the radiator, the fan worked perfectly, but when I put it on top of the engine the fan ran all the time. I fixed this by making a little shield for the sensor so that it was protected from the heat of the engine bay. I also found that the fan worked OK one day and not the next. This was because the current draw of the fan motor was too great for the transistor, which was failing under load. I fixed this by adding a relay to control the fan motor. This made the fan work correctly when it was required to cool down the engine (see annotated photographs of this).*  *The above expected learner responses are indicative only and relate to just part of what is required.* | The learner efficiently implements basic interfacing procedures in the sensor-controlled cooling fan system by:   * choosing appropriate component types and values for the interface   For example:  *I found that the Thermistor has a resistance of 68 kΩ when the temperature was at 100 degrees Celsius and 4 kΩ when the temperature was 0 degrees Celsius. So I chose a resistor in between these two values to use in the voltage divider. I used a 30 kΩ resistor.*   * writing well-structured and clearly annotated interface software   For example:  See Appendix 3 for an example illustrating different versions of a code fragment.   * using data sheets or calculations to assist in choosing appropriate component types and values for the interface   For example:  From data sheet, diode is rated at 3V and 20mA.   * testing and debugging a functional model of the interface for the sensor-controlled cooling fan   For example:  *I found that when the sensor was placed on top of the radiator, the fan worked perfectly, but when I put it on top of the engine the fan ran all the time. I fixed this by making a little shield for the sensor so that it was protected from the heat of the engine bay. I also found that the fan worked OK one day and not the next. This was because the current draw of the fan motor was too great for the transistor, which was failing under load. I fixed this by adding a relay to control the fan motor. This made the fan work correctly when it was required to cool down the engine (see annotated photographs of this).*  *The above expected learner responses are indicative only and relate to just part of what is required.* |

Final grades will be decided using professional judgement based on an examination of the evidence provided against the criteria in the Achievement Standard. Judgements should be holistic, rather than based on a checklist approach.

### Appendix 1: Learner evidence for Achieved (Example only. The code shown below is meant to be reflective of the depth and annotation required only, to provide guidance on the depth of code expected at each level and does not relate to the context of the task.)

The code is all jammed onto consecutive lines, which does little to make the structure clear (even though it is essentially the same code as in the Excellence example). Labels are not used and comments are non-existent. Even the descriptor at the start does not really capture the essence of what the barrier arm does. The code, while correct, is not that straightforward or easy to follow.

/\* This programme raises a barrier arm when a car is detected. \*/

int sensorValue = 0;

int Switch\_value = 0;

void setup()

pinMode(13, OUTPUT);

pinMode (0, INPUT);

pinMode (8, INPUT);

}

void loop() {

while (Switch\_value = = LOW)

Switch\_value = analogRead(8);

digitalWrite(13, HIGH);

while (Sensor\_value < 500)

Sensor\_value = analogRead(0);

digitalWrite(13, LOW);

}

### Appendix 2: Learner evidence for Merit (Example only. The code shown below is meant to be reflective of the depth and annotation required only, to provide guidance on the depth of code expected at each level and does not relate to the context of the task.)

This piece of code is well structured, but not clearly annotated. Annotations are there, but they are not helpful in determining what tasks the code is addressing. The comprehensibility of the code is compromised by a failure to use labels that give an intuitive understanding of the functions of the input and output pins on the microcontroller.

The comments highlighted in yellow are not part of what a learner might write. They are comments explaining where the annotation is incomplete.

/\* This piece of code waits until a switch is pressed. It then turns on a motor to raise a barrier arm.

A light sensor detects when the barrier arm reaches the ‘up’ position and the motor stops automatically.\*/

int arm\_up = 13; //create expression ‘arm\_up’ to refer to pin 13 where motor is connected

int sensor = 0; // Create expression ‘sensor’ to refer to pin 0 where sensor is connected

int Switch = 8; //Create expression ‘Switch’ to digital pin 8 where switch is connected

int sensorValue = 0; // variable to store the value coming from the voltage divider

int Switch\_value = 0;

void setup() {

pinMode(13, OUTPUT); // declare the pin (13) as an output but no indication of what pin 13 //does

pinMode (0, INPUT); //Declare pin 0 as an analogue input.

pinMode (8, INPUT); //Declare the pin as an input, but no indication what it’s for

}

void loop() {

while (Switch\_value = = LOW) //

Switch\_value = analogRead(8); // read pin 8 but no indication what information this provides

digitalWrite(13, HIGH); //Write pin 13 high, but no indication what this is doing

while (Sensor\_value < 500) //while light beam not blocked

Sensor\_value = analogRead(0); //just keep reading the sensor pin

digitalWrite(13, LOW); //Write pin 13 LOW, but no indication what this controls

}

### Appendix 3: Learner evidence for Excellence (Example only. The code shown below is meant to be reflective of the depth and annotation required only, to provide guidance on the depth of code expected at each level and does not relate to the context of the task.)

This piece of code is clearly written, well annotated, and well structured.

Yellow: Examples of given programme structures; others include “if” and “if-else” statements.

Green: Examples of creating labels that make the code more readily understood.

/\* This piece of code waits until a switch is pressed. It then turns on a motor to raise a barrier arm.

A light sensor detects when the barrier arm reaches the ‘up’ position and the motor stops automatically.\*/

int arm\_up = 13; //create expression ‘arm\_up’ to refer to pin 13 where motor is connected

int sensor = 0; // Create expression ‘sensor’ to refer to pin 0 where sensor is connected

int Switch = 8; //Create expression ‘Switch’ to digital pin 8 where switch is //connected

int Sensor\_value = 0; // variable to store the value coming from the voltage divider

int Switch\_value = 0;

void setup() {

pinMode(arm\_up, OUTPUT); // declare the arm\_up pin (13) as an output

pinMode (sensor, INPUT); //Declare the sensor pin as an analogue input

pinMode (Switch, INPUT); //Declare the switch pin as a digital input

}

void loop() {

while (Switch\_value == LOW) //while light beam not blocked

Switch\_value = analogRead(Switch); // keep reading the switch pin:

digitalWrite(arm\_up, HIGH); //Once beam is blocked, start raising arm

while (Sensor\_value < 500) //while light beam not blocked

Sensor\_value = analogRead(sensor); //just keep reading the sensor pin

digitalWrite(arm\_up, LOW); //once beam is blocked, stop arm from rising

}