基於 AUTOSAR CLASSIC 的 PORT DRIVER 開發

指導教授：張大緯
專題成員：曾宇弘

目錄

研究動機

研究動機

汽車產業中大量地使用嵌入式系統，但卻苦無一個標準的架構規範，使系統的維護、升級與更換都遇上了不小的困難。因此陸續出現了使汽車系統標準化的協議。

為了能使系統更方便的去維護與設定，本專題將會基於 AUTOSAR Classic 架構，在現有的 OSEK/VDX 的車用作業系統 MilkShopOS 之下，進行 Port Driver 的開發。

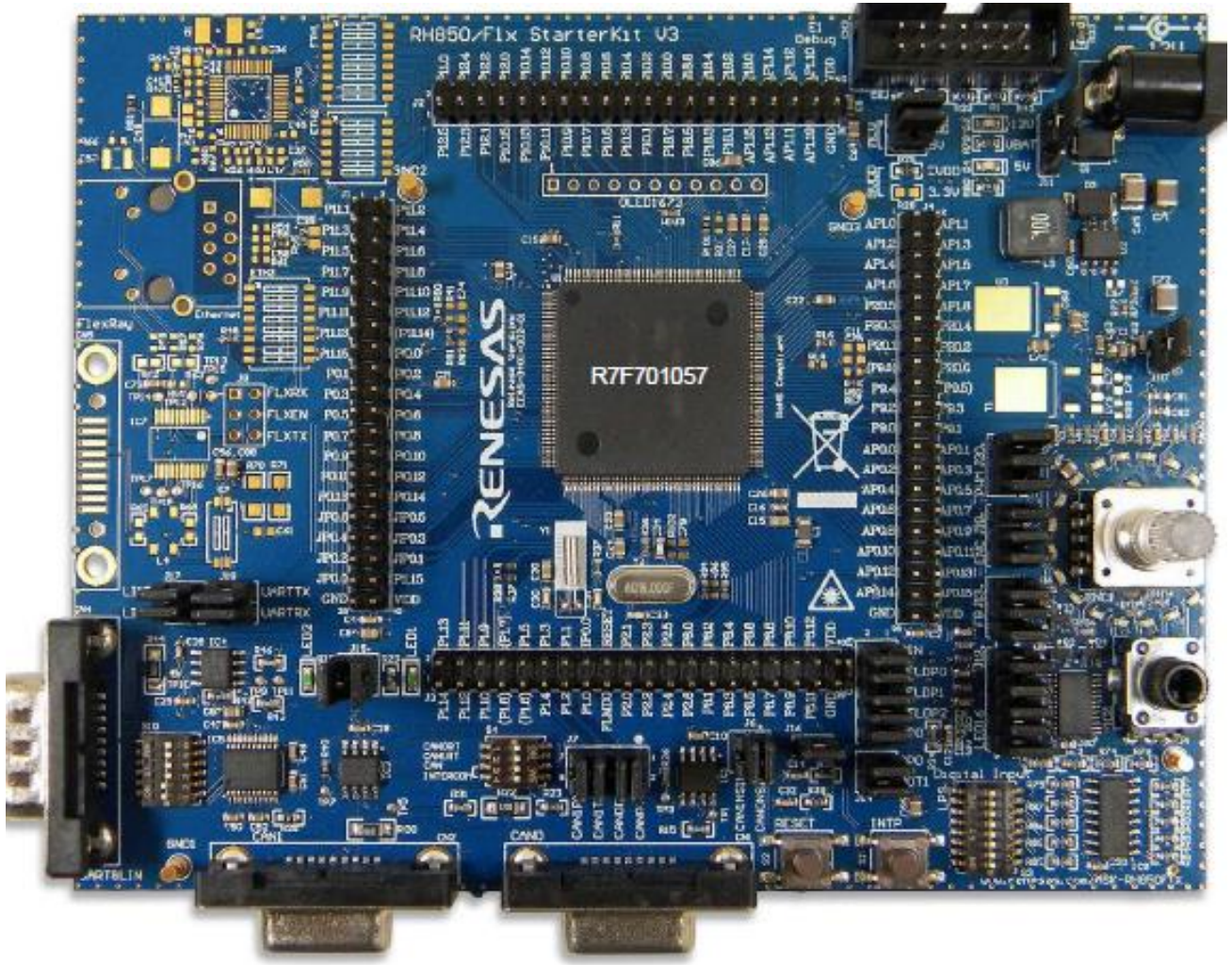
開發工具

開發工具

RH850/F1L

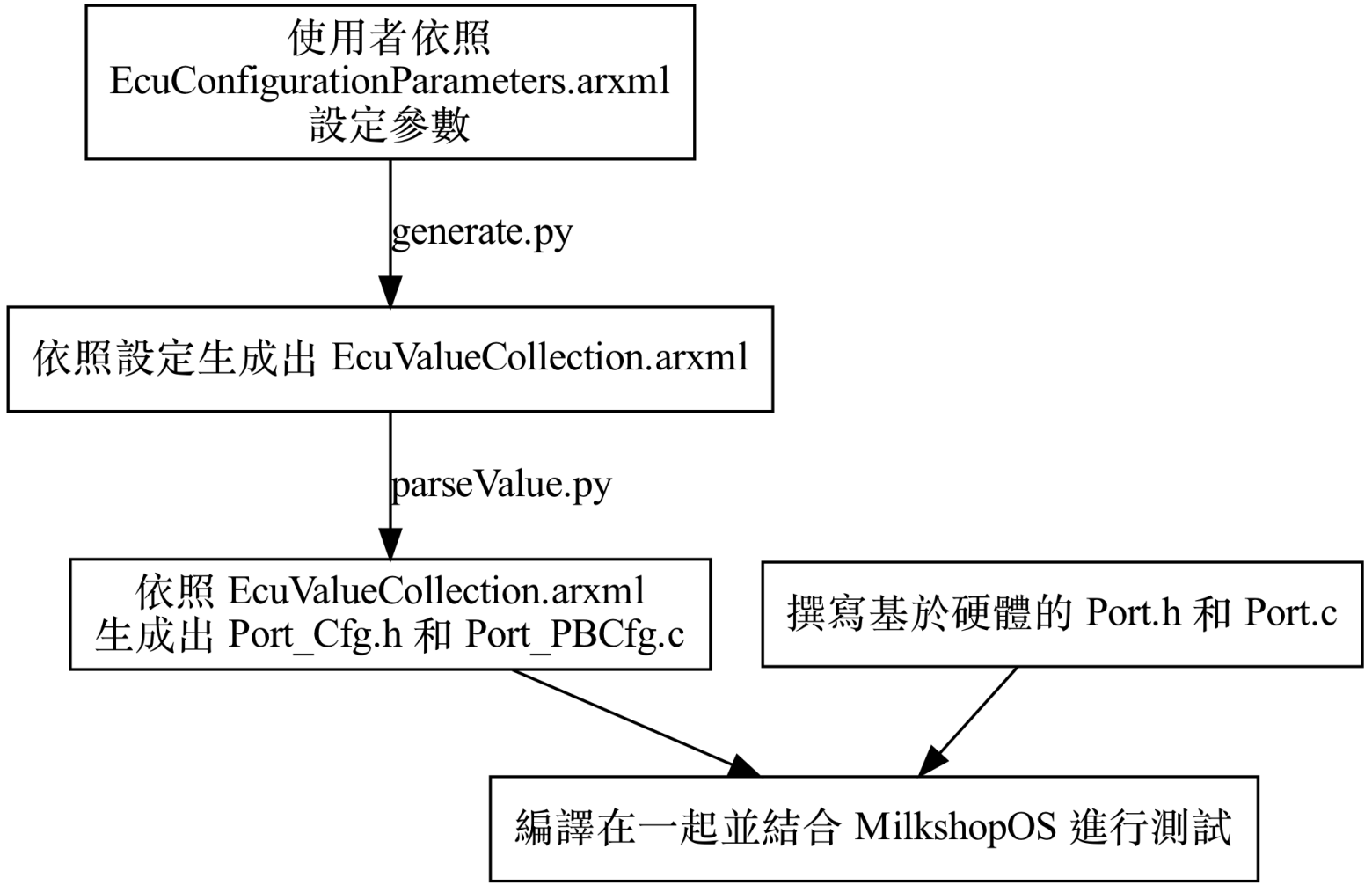
Python 3.9.0

CS+ for CC



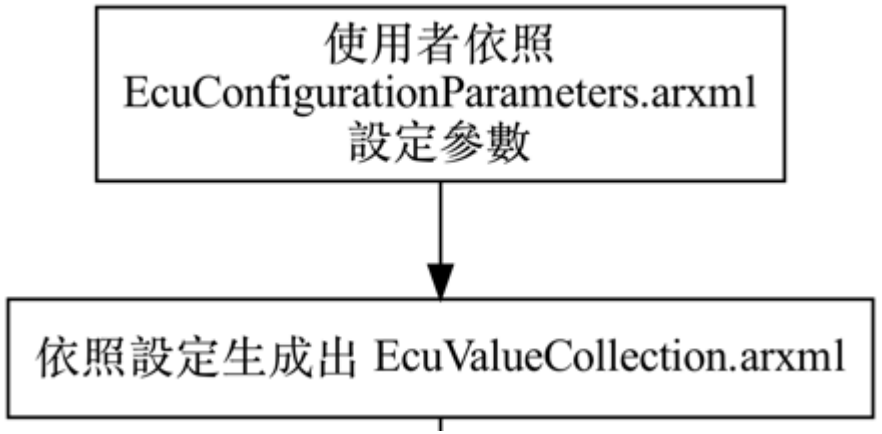
研究流程

研究流程


使用者依照
EcuConfigurationParameters.arxml
設定參數
generate.py
依照設定生成出 EcuValueCollection.arxml
parseValue.py
依照 EcuValueCollection.arxml
生成出 Port_Cfg.h 和 Port_PBCfg.c
撰寫基於硬體的 Port.h 和 Port.c
編譯在一起並結合 MilkshopOS 進行測試

研究方法

研究方法



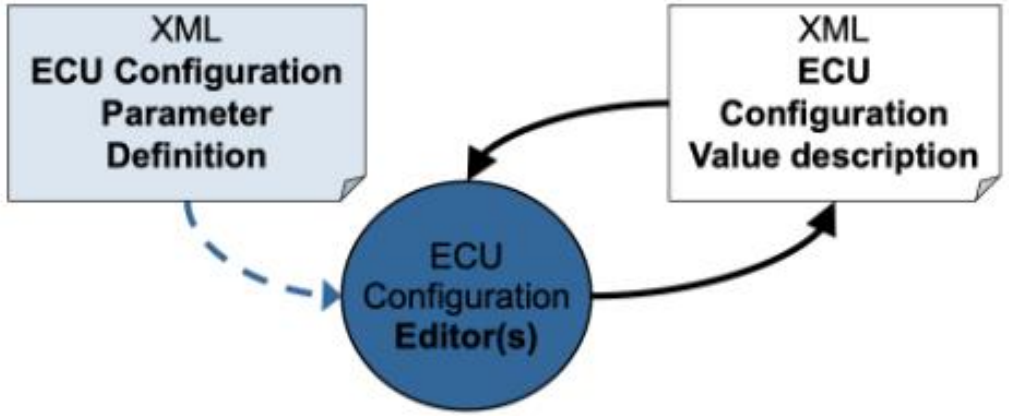


Figure 2.1: Parameter Definition and ECU Configuration Value files

EcuConfigurationParameters.arxml (右圖)

```
<!-- PARAMETER DEFINITION: PortPinLevelValue -->
<ECUC-ENUMERATION-PARAM-DEF UUID="ECUC:83fdd686-ae98-423f-bdfb-c46dad6783f4">
  <SHORT-NAME>PortPinLevelValue</SHORT-NAME>
  <DESC>
    <L-2 L="EN">Port Pin Level value from Port pin list.</L-2>
  </DESC>
  <RELATED-TRACE-ITEM-REF BASE="ArTrace" DEST="TRACEABLE">ECUC_Port_00129</RELATED-TRACE-ITEM-REF>
  <LOWER-MULTIPLICITY>1</LOWER-MULTIPLICITY>
  <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
  <SCOPE>LOCAL</SCOPE>
  <ORIGIN>AUTOSAR_ECUC</ORIGIN>
  <POST-BUILD-VARIANT-VALUE>true</POST-BUILD-VARIANT-VALUE>
  <VALUE-CONFIG-CLASSES>
    <ECUC-VALUE-CONFIGURATION-CLASS>
      <CONFIG-CLASS>POST-BUILD</CONFIG-CLASS>
      <CONFIG-VARIANT>VARIANT-POST-BUILD</CONFIG-VARIANT>
    </ECUC-VALUE-CONFIGURATION-CLASS>
    <ECUC-VALUE-CONFIGURATION-CLASS>
      <CONFIG-CLASS>PRE-COMPILE</CONFIG-CLASS>
      <CONFIG-VARIANT>VARIANT-PRE-COMPILE</CONFIG-VARIANT>
    </ECUC-VALUE-CONFIGURATION-CLASS>
  </VALUE-CONFIG-CLASSES>
  <SYMBOLIC-NAME-VALUE>false</SYMBOLIC-NAME-VALUE>
  <LITERALS>
    <ECUC-ENUMERATION-LITERAL-DEF UUID="ECUC:049ec56c-bfe5-853f-310c-f70254daa840">
      <SHORT-NAME>PORT_PIN_LEVEL_HIGH</SHORT-NAME>
      <ORIGIN>AUTOSAR_ECUC</ORIGIN>
    </ECUC-ENUMERATION-LITERAL-DEF>
    <ECUC-ENUMERATION-LITERAL-DEF UUID="ECUC:dea65423-5596-83dc-2604-8a2134878b00">
      <SHORT-NAME>PORT_PIN_LEVEL_LOW</SHORT-NAME>
      <ORIGIN>AUTOSAR_ECUC</ORIGIN>
    </ECUC-ENUMERATION-LITERAL-DEF>
  </LITERALS>
</ECUC-ENUMERATION-PARAM-DEF>
```

EcuValueCollection.arxml (下圖)

```
<ECUC-TEXTUAL-PARAM-VALUE>
  <DEFINITION-REF DEST="ECUC-ENUMERATION-PARAM-DEF">/AUTOSAR/EcucDefs/Port/PortConfigSet/PortContainer/PortPin/PortPinLevelValue</DEFINITION-REF>
  <VALUE>PORT_PIN_LEVEL_HIGH</VALUE>
</ECUC-TEXTUAL-PARAM-VALUE>
```

研究方法

依照 EcuValueCollection.arxml
生成出 Port_Cfg.h 和 Port_PBCfg.c

```
<SHORT-NAME>PortContainer1</SHORT-NAME>
<DEFINITION-REF DEST="ECUC-PARAM-CONF-CONTAINER-DEF">/AUTOSAR/EcucDefs/Port/PortConfigSet/PortContainer</DEFINITION-REF>
<PARAMETER-VALUES>
  <ECUC-NUMERICAL-PARAM-VALUE>
    <DEFINITION-REF DEST="ECUC-INTEGER-PARAM-DEF">/AUTOSAR/EcucDefs/Port/PortConfigSet/PortContainer/PortNumberOfPortPins</DEFINITION-REF>
    <VALUE>1</VALUE>
  </ECUC-NUMERICAL-PARAM-VALUE>
</PARAMETER-VALUES>
<SUB-CONTAINERS>
  <ECUC-CONTAINER-VALUE>
    <SHORT-NAME>P12_5</SHORT-NAME>
    <PARAMETER-VALUES>
      <ECUC-TEXTUAL-PARAM-VALUE>
        <DEFINITION-REF DEST="ECUC-ENUMERATION-PARAM-DEF">/AUTOSAR/EcucDefs/Port/PortConfigSet/PortContainer/PortPin/PortPinDirection</DEFINITION-REF>
        <VALUE>PORT_PIN_OUT</VALUE>
      </ECUC-TEXTUAL-PARAM-VALUE>
      <ECUC-NUMERICAL-PARAM-VALUE>
        <DEFINITION-REF DEST="ECUC-BOOLEAN-PARAM-DEF">/AUTOSAR/EcucDefs/Port/PortConfigSet/PortContainer/PortPin/PortPinDirectionChangeable</DEFINITION-REF>
        <VALUE>1</VALUE>
      </ECUC-NUMERICAL-PARAM-VALUE>
      <ECUC-NUMERICAL-PARAM-VALUE>
        <DEFINITION-REF DEST="ECUC-INTEGER-PARAM-DEF">/AUTOSAR/EcucDefs/Port/PortConfigSet/PortContainer/PortPin/PortPinId</DEFINITION-REF>
        <VALUE>17</VALUE>
```

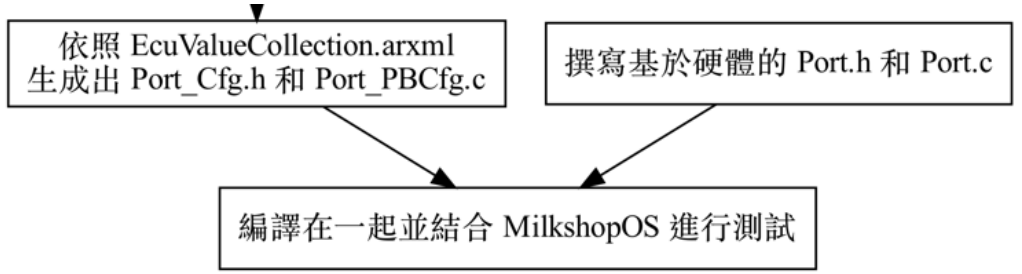
```
#define PortConf_PortContainer1_P12_5 17
```

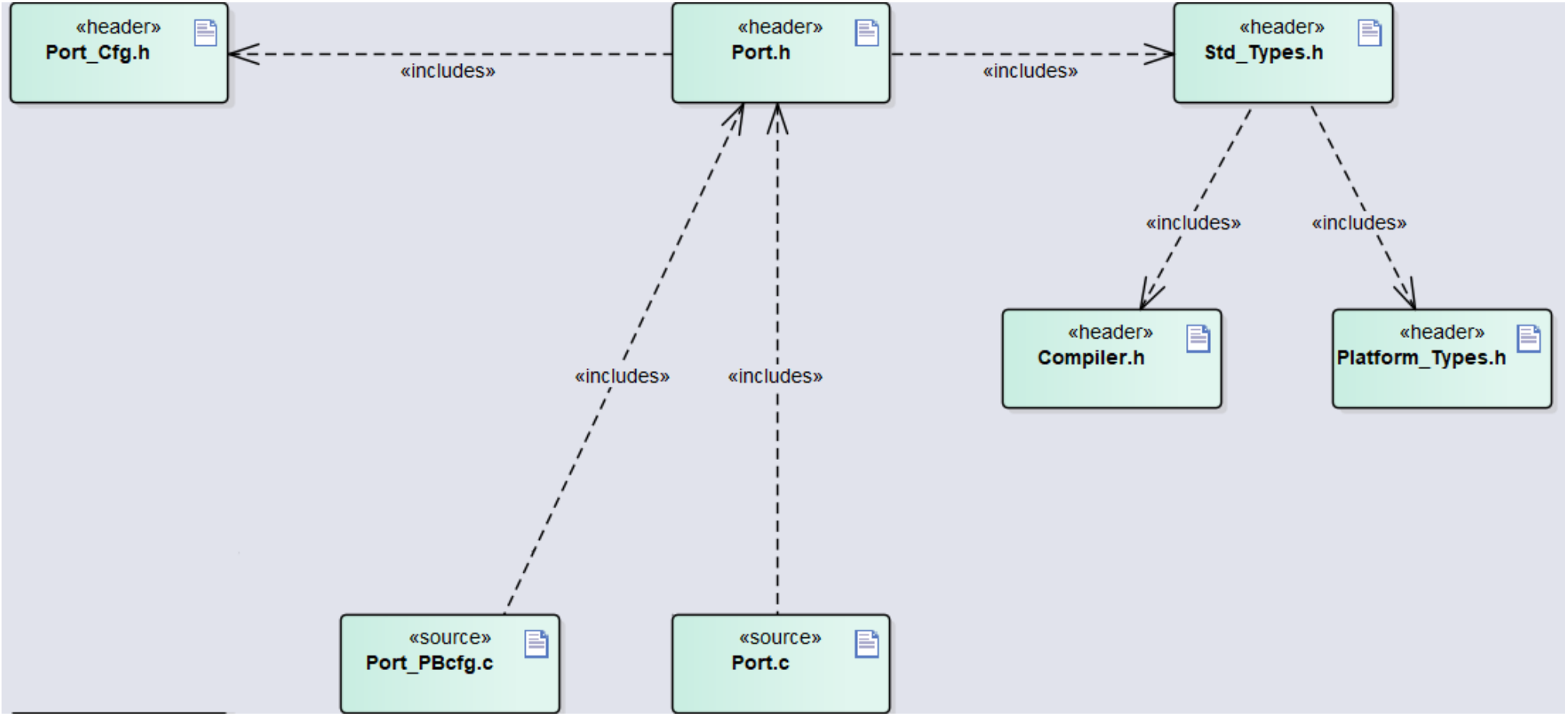
研究方法

撰寫基於硬體的 Port.h 和 Port.c

```c
/* Function definition */

void Port_Init(const Port_ConfigType* ConfigPtr);

#if PORT_SET_PIN_DIRECTION_API == STD_ON
void Port_SetPinDirection(Port_PinType Pin, Port_PinDirectionType Direction);
#endif

void Port_RefreshPortDirection(void);

#if PORT_VERSION_INFO_API == STD_ON
void Port_GetVersionInfo(Std_VersionInfoType *versioninfo);
#endif

#if PORT_SET_PIN_MODE_API == STD_ON
void Port_SetPinMode(Port_PinType Pin, Port_PinModeType Mode);
#endif
```

研究方法

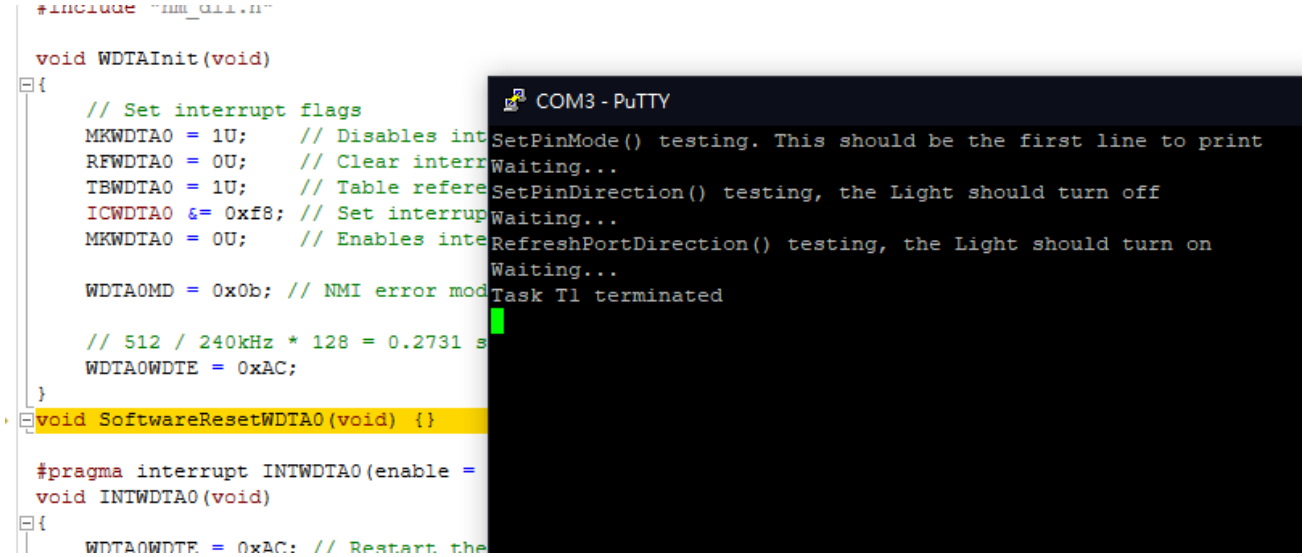

依照 EcuValueCollection.arxml
生成出 Port_Cfg.h 和 Port_PBCfg.c
撰寫基於硬體的 Port.h 和 Port.c
編譯在一起並結合 MilkshopOS 進行測試


«header»
Port_Cfg.h
«includes»
«header»
Port.h
«includes»
«header»
Std_Types.h
«includes»
«includes»
«includes»
«includes»
«header»
Compiler.h
«header»
Platform_Types.h
«source»
Port_PBcfg.c
«source»
Port.c

成果

成果

```
INFO: list of configuration files:
INFO: configuration file 1: application/osek.oil
INFO: list of files to be generated:
INFO: generated file 1: gen/boot.asm.php
INFO: output directory: device
INFO: reading application/osek.oil
INFO: generating gen/boot.asm.php to device/boot.asm
INFO: Generation Finished with WARNINGS: 0 and ERRORS: 0
cd port_driver && python3 generate.py
Generate EcuValueCollection.arxml Success !
cd port_driver && python3 parseValue.py
Generate Port_Cfg.h and Port_PBCfg.c Success !
PS C:\Users\Mickey\Downloads\MilkshopOSEK\MilkshopOSEK\MilkshopOSEK>
```



感謝聆聽