# USER’S REFERENCE MANUAL

# DIO64-ARD

## 64-Bit Digital I/O
## Shield for Arduino and Compatibles

| | | |
|---|---|---|
| Model No. | 100-7728 | |
| Doc. No. | M7728 | Rev: 1.00 5/11/22 |

SCIDYNE®

649 School Street / Pembroke, MA  02359 USA / Tel:  (781) 293-3059
www.scidyne.com

## DISCLAIMER:

This document contains proprietary information regarding SCIDYNE and its products. The information is subject to change without notice. SCIDYNE makes no warranty of any kind with regard to this material, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. SCIDYNE shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material. No part of this document may be duplicated in any form without prior written consent of SCIDYNE.

## WARRANTY:

SCIDYNE warrants this product against defects in materials and workmanship and, that it shall conform to specifications current at the time of shipment, for a period of one year from date of shipment. Duration and conditions of warranty may be superseded when the product is integrated into other SCIDYNE products. During the warranty period, SCIDYNE will, at its option and without charge to Buyer, either repair or replace products which prove defective. Repair or replacement of a defective product or part thereof does not extend the original warranty period.

## WARRANTY SERVICE:

For warranty service or repair, this product must be returned to a service facility designated by SCIDYNE. The Buyer must obtain prior approval and a Return Material Authorization (RMA) number before returning any products. The RMA number must be clearly visible on the shipping container. The Buyer shall prepay shipping and insurance charges to the service facility and SCIDYNE shall pay shipping and insurance charges to Buyer's facility for products repaired or replaced. SCIDYNE may, at its discretion, bill the Buyer for return shipping and insurance charges for products received for repair but determined to be non-defective. Additionally, the Buyer shall pay all shipping charges, duties and taxes for products returned to SCIDYNE from another country.

## LIMITATION OF WARRANTY:

The forgoing warranty shall not apply to defects resulting from improper or negligent maintenance by the Buyer, Buyer-supplied products or interfacing, unauthorized modifications or misuse, operation outside the published specifications of the product or improper installation site preparation or maintenance, or the result of an accident. The design and implementation of any circuit using this product is the sole responsibility of the Buyer. SCIDYNE does not warrant the Buyer's circuitry or malfunctions of SCIDYNE products that result from the Buyer's circuitry. In addition, SCIDYNE does not warrant any damage that occurs as a result of the Buyer's circuit or any defects that result from Buyer-supplied products. This Warranty does not cover normal preventative maintenance items such as fuse replacement, lamp replacement, resetting of circuit breakers, cleaning of the Product or problems caused by lack of preventative maintenance, improper cleaning, improper programming or improper operating procedures. No other warranty is expressed or implied. SCIDYNE specifically disclaims the implied warranties of merchantability and fitness for a particular purpose. Some states do not permit limitation or exclusion of implied warranties; therefore, the aforesaid limitation(s) or exclusion(s) may not apply to the Buyer. This warranty gives you specific legal rights and you may have other rights which vary from state to state.

## CERTIFICATION:

Testing and other quality control techniques are utilized to the extent SCIDYNE deems necessary to support this warranty. Specific testing of all parameters is not necessarily performed, except those mandated by government requirements.

## 30-DAY PRODUCT EVALUATION POLICY:

SCIDYNE offers a no-risk trial for initial, low quantity, evaluation purchases. Items purchased for evaluation can be returned within 30 days of purchase for a full refund less shipping charges. The Buyer must obtain a Return Material Authorization (RMA) number before returning any products. The entire package, including hardware, software, documentation, discount coupons and any other accessories supplied must be returned intact and in new and working condition. This policy will not be honored for packages that are not returned complete and intact. The Buyer shall prepay shipping and insurance charges to SCIDYNE. To expedite the return process, the RMA number must be clearly visible on the shipping container. SCIDYNE will cancel the invoice, refund by check or issue credit to your credit card within 10 days after receipt of returned merchandise.

## LIFE SUPPORT POLICY:

Certain applications may involve the risks of death, personal injury or severe property or environmental damage ("Critical Applications").

SCIDYNE products are not designed, intended, authorized or warranted to be suitable for use in life-support applications, devices or systems or other critical applications without the express written approval of the president of SCIDYNE.

# Table of Contents

# Conventions and Terminology used in this publication

## Safety and Usage Conventions

**Note:** *Provides important information and useful tips that will assist in the understanding and operation of this product.*

**Caution:** *Calls attention to a procedure, practice, or condition that could possibly cause equipment damage or bodily injury.*

**Danger:** ***Calls attention to a procedure, practice, or condition that is likely to cause extensive equipment damage, severe bodily injury, or death if not observed.***

## Terminology

### Logic Conditions

Unless otherwise noted, logic signals are designated as TRUE (Set) and FALSE (Clear). Names with an asterisk (*) postscript or overlined are inverted or active low. Unless otherwise noted TRUE is considered logic '1' (Positive Voltage, +5Vdc or +3.3Vdc) and FALSE is considered logic '0' (0Vdc).

### Numbering Systems

Computerized equipment often requires its numeric data to be represented in different forms depending on the audience and information being conveyed. Decimal numbers are typically used for end-user data entry and display while internally these values are converted and manipulated in native binary. Hexadecimal numbers are often used by programmers as an intermediate level between binary and decimal notations.

| Base | Name | Format (MS ←--→ LS) |
|---|---|---|
| 2 | Binary | 0b10111001 or 1011 1001$_2$ |
| 10 | Decimal | 185 |
| 16 | Hexadecimal | 0xB9 or B9$_{16}$ or HB9 |

### Multi-Byte Word Formats

In this document multi-byte values are shown as 0x1234 where 12 represents the most-significant byte and 34 is the least significant byte. Depending on your particular system the values could be internally stored as little-endian or big-endian.

# Introduction

The DIO64-ARD is an Arduino compatible peripheral board designed to satisfy general digital Input/Output requirements in a broad range of embedded applications. The hardware has been engineered to operate at either 3.3V or 5V making it compatible with common microcontroller boards such as Arduino Uno, Mega, Due, Adafruit Metro, SparkFun Red Board, and numerous others.

*This manual provides basic insight of the hardware and fundamental software concepts needed to apply the DIO64-ARD. It is not intended to be a comprehensive resource. The reader is encouraged to refer to the actual chip manufacturers data sheets for detailed understanding of the features and capabilities of the devices used.*

## Key Features

- 64 Digital Input / Output channels via eight 8-bit Ports
  - – Industry Standard MCP23017 Chips
  - – All channels are Bi-Directional
  - – Programable pull-up resistors
  - – Change-of-State and pattern matching interrupt capability
- High-Speed I$^2$C interface, 100kHz, 400kHz, 1.7Mbps speed
- Optional Stemma QT / Qwiic and nodeLynk connectors for driving external hardware
- Use with libraries from Adafruit, SparkFun, and others
- Selectable 3.3V or 5V operating voltage
- Low Power requirement
- Standard Arduino R3 footprint

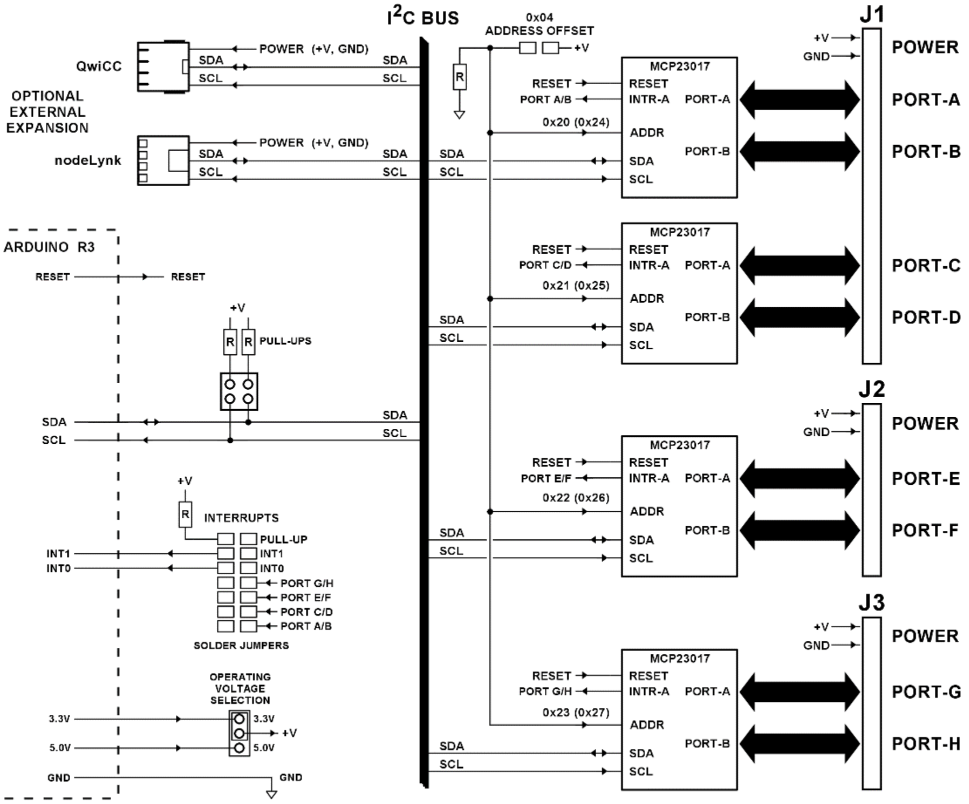


**Figure 1 –** DIO64-ARD Simplified Block Diagram

## Component Identification

To properly apply the DIO64-ARD it is necessary to become familiar with its various components. The following figure and accompanying table briefly describe their functions and locations. Subsequent sections of this manual explain their purpose in greater detail.

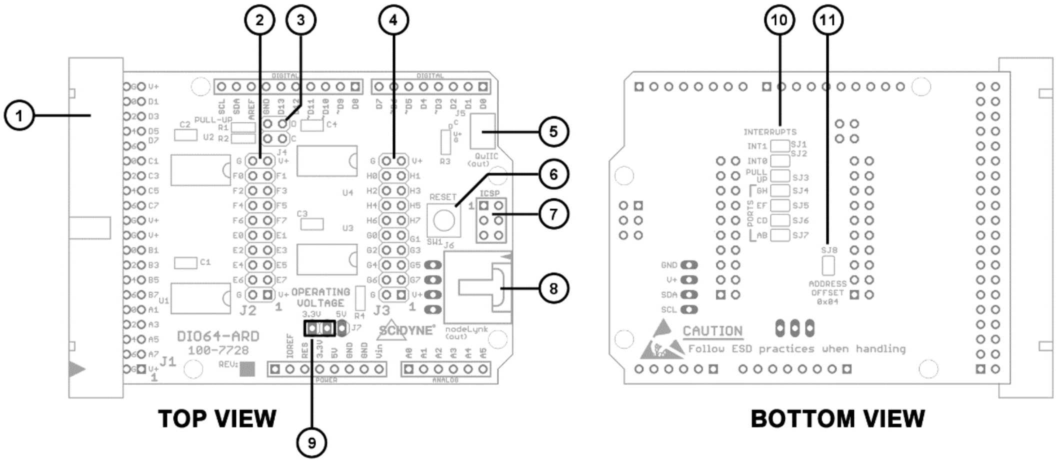


**Figure 2 –** DIO64-ARD Component Identification

1 **I/O connector (J1)**
This 40-pin IDC header connects external devices to PORT-A, PORT-B, PORT-C, and PORT-D. Please refer to Appendix-A for wiring information.

2 **I/O connector (J2)**
This 20-pin header connects external devices to PORT-E and PORT-F. Please refer to Appendix-A for wiring information.

3 **I$^2$C Pull-Ups (J4)**
These jumpers enable optional Pull-Up resistors on the SDA and SCL signals. Only one set of Pull-Up resistors should be used on any I$^2$C bus segment and generally at the furthest point. The Pull-Ups are referenced to V+ as determined by J7.

4 **I/O connector (J3)**
This 20-pin header connects external devices to PORT-G and PORT-H. Please refer to Appendix-A for wiring information.

5 **Optional Qwiic Connector (J5)**
This connector allows standard Stemma QT / Qwiic[1] devices to be connected through the MUTI-IO-ARD board.

6 **RESET Push Button (SW1)**
Momentarily pressing this button will reset the entire Arduino system.

7 **Optional ICSP Stackthrough connector**
An optional 2x3 header can be installed to pass ICSP signals from a lower board to one attached above.

8 **Optional nodeLynk Connector (J6)**
This connector allows external device supporting the nodeLynk[1] interface to be driven by the DIO64-ARD.

9 **I/O Voltage Selection (J7)**
The operating voltage for the DIO64-ARD is set by this jumper. The default position is 3.3V.
**Note: This setting MUST match the operating voltage of the device the DIO64-ARD is plugged in to. Failure to do so could damage both boards.**

10 **Interrupt Configuration Solder Jumpers (SJ1 – SJ7)**
This jumper block sets which interrupt sources will be used by the DIO64-ARD to request interrupt service from the host.

11 **Address Offset Solder Jumper (SJ8)**
This jumper block determines the base address of the Digital I/O chips on the I$^2$C bus.

1) Stemma QT / Qwicc and nodeLynk are peripheral hardware buses based on I$^2$C communications. Popularized by Adafruit, SparkFun, National Control Devices and other third parties.

# Board Usage Details

The DIO64-ARD module uses four industry standard MCP23017 I/O Expander chips to provide 64 non-isolated digital Input/Output channels across eight 8-bit ports. This device is very versatile and offers flexible configurations, including software programable port direction and interrupt-driven change-of-state and pattern matching functions. Each channel features TTL/CMOS compatible signal levels and up to ±25mA drive capability. In addition, software programmable weak pull-up resistors to V+ are available on any of the input channels. This feature makes sensing open-collector, switches, and contact-closure type devices simple and straight-forward. All channels default to inputs during system reset.

External devices attach to the DIO64-ARD through pins of connectors J1, J2, and J3. Their pinouts and their relationship to the four MCP23017 chips is show below.

Companion terminal boards, SCIDYNE PN 100-7625/40 and 100-7625/20, are available to make field wiring easier.

J1[(2)]

| Source | | J1 | | | Source |
|---|---|---|---|---|---|
| | V+ [1] | 1▼ | 2 | Ground | |
| U1, PortA.7 | PA7 | 3 | 4 | PA6 | U1, PortA.6 |
| U1, PortA.5 | PA5 | 5 | 6 | PA4 | U1, PortA.4 |
| U1, PortA.3 | PA3 | 7 | 8 | PA2 | U1, PortA.2 |
| U1, PortA.1 | PA1 | 9 | 10 | PA0 | U1, PortA.0 |
| U1, PortB.7 | PB7 | 11 | 12 | PB6 | U1, PortB.6 |
| U1, PortB.5 | PB5 | 13 | 14 | PB4 | U1, PortB.4 |
| U1, PortB.3 | PB3 | 15 | 16 | PB2 | U1, PortB.2 |
| U1, PortB.1 | PB1 | 17 | 18 | PB0 | U1, PortB.0 |
| | V+ [1] | 19 | 20 | Ground | |
| | V+ [1] | 21 | 22 | Ground | |
| U2, PortA.7 | PC7 | 23 | 24 | PC6 | U2, PortA.6 |
| U2, PortA.5 | PC5 | 25 | 26 | PC4 | U2, PortA.4 |
| U2, PortA.3 | PC3 | 27 | 28 | PC2 | U2, PortA.2 |
| U2, PortA.1 | PC1 | 29 | 30 | PC0 | U2, PortA.0 |
| U2, PortB.7 | PD7 | 31 | 32 | PD6 | U2, PortB.6 |
| U2, PortB.5 | PD5 | 33 | 34 | PD4 | U2, PortB.4 |
| U2, PortB.3 | PD3 | 35 | 36 | PD2 | U2, PortB.2 |
| U2, PortB.1 | PD1 | 37 | 38 | PD0 | U2, PortB.0 |
| | V+ [1] | 39 | 40 | Ground | |

J2[(2)]

| Source | | J2 | | | Source |
|---|---|---|---|---|---|
| | V+ [1] | 1▼ | 2 | Ground | |
| U3, PortA.7 | PE7 | 3 | 4 | PE6 | U3, PortA.6 |
| U3, PortA.5 | PE5 | 5 | 6 | PE4 | U3, PortA.4 |
| U3, PortA.3 | PE3 | 7 | 8 | PE2 | U3, PortA.2 |
| U3, PortA.1 | PE1 | 9 | 10 | PE0 | U3, PortA.0 |
| U3, PortB.7 | PF7 | 11 | 12 | PF6 | U3, PortB.6 |
| U3, PortB.5 | PF5 | 13 | 14 | PF4 | U3, PortB.4 |
| U3, PortB.3 | PF3 | 15 | 16 | PF2 | U3, PortB.2 |
| U3, PortB.1 | PF1 | 17 | 18 | PF0 | U3, PortB.0 |
| | V+ [1] | 19 | 20 | Ground | |

J3[(2)]

| Source | | J3 | | | Source |
|---|---|---|---|---|---|
| | V+ [1] | 1▼ | 2 | Ground | |
| U4, PortA.7 | PG7 | 3 | 4 | PG6 | U4, PortA.6 |
| U4, PortA.5 | PG5 | 5 | 6 | PG4 | U4, PortA.4 |
| U4, PortA.3 | PG3 | 7 | 8 | PG2 | U4, PortA.2 |
| U4, PortA.1 | PG1 | 9 | 10 | PG0 | U4, PortA.0 |
| U4, PortB.7 | PH7 | 11 | 12 | PH6 | U4, PortB.6 |
| U4, PortB.5 | PH5 | 13 | 14 | PH4 | U4, PortB.4 |
| U4, PortB.3 | PH3 | 15 | 16 | PH2 | U4, PortB.2 |
| U4, PortB.1 | PH1 | 17 | 18 | PH0 | U4, PortB.0 |
| | V+ [1] | 19 | 20 | Ground | |

Notes:

1) V+ = 3.3V or 5V depending on setting of jumper J7. Supplied by Host. Non-Isolated and Unfused.
2) Orientation shown for clarity. Actual connectors are rotated 180 degrees on circuit board.

## Operating Voltage

The DIO64-ARD circuitry can operate at either 3.3V or 5V (i.e.; V+). This ability allows its use with a wide variety of microcontroller boards and compatible hardware including those based on FPGAs. The operating voltage determines the logic levels of the signals between the DIO64-ARD and the host, as well as the logic levels of any external circuitry connected through connectors J1, J2, J3, J5, and J6.

Place jumper J7 to either the 3.3V or 5V position to set the V+ operating voltage.

*The operating I/O voltage of the DIO64-ARD must be properly configured to match that of the host microcontroller board. A mismatch could permanently damage the DIO64-ARD, the microcontroller board, or both devices.*

*If relocating the DIO64-ARD to a different system this prerequisite must be verified and the J7 Operating Voltage jumper changed if necessary.*

*The source of the 3.3V and 5V operating power is derived from the host.*
*The DIO64-ARD does not possess any on-board power supply regulation or current limiting circuitry.*

## I$^2$C Addresses

The DIO64-ARD use four consecutive addresses on the I$^2$C bus as summarized in the table below. The default addresses can be offset 0x04 by soldering the pads of SJ8 (located on back of board).

*To prevent conflicts, each device residing within the 0x20 – 0x27 address range must have a unique address on the I$^2$C bus segment they share. Using Addrerss Offset helps avoid address conflicts allowing two DIO64-ARD boards, or other devices with similar addresses, to be used within the same system.*

| I$^2$C Addresses used by the DIO64-ARD | | | | |
|---|---|---|---|---|
| Device | I$^2$C Address (hex) | | Circuit Function | Associated Connector |
| | Default Address | Address Offset (SJ8 enabled) | | |
| U1, MCP23017 | 0x20 | 0x24 | PORTA[7..0], PORTB[7..0] | J1 |
| U2, MCP23017 | 0x21 | 0x25 | PORTC[7..0], PORTD[7..0] | |
| U3, MCP23017 | 0x22 | 0x26 | PORTE[7..0], PORTF[7..0] | J2 |
| U4, MCP23017 | 0x23 | 0x27 | PORTG[7..0], PORTH[7..0] | J3 |

## Pull-Up Resistors

Jumper block J4 enables optional 4.7K Pull-Up resistors on the I$^2$C SDA and SCL signals to the V+ operating voltage. The use of Pull-Up resistors is often required to achieve higher I$^2$C bus speeds. Only one set of Pull-Ups should be enabled within the I$^2$C bus and typically at the furthest physical point.

## Interrupts

The DIO64-ARD can optionally generate an interrupt whenever a digital input has changed state or when a pre-defined pattern appears on digital inputs.  This capability is part of the MCP23017 chips and DIO64-ARD circuitry.

Each MCP23017 chip has two interrupt outputs, INTA and INTB, corresponding to the chips PORTA and PORTB. However, only the INTA signal (of each MCP23017 chip) is wired to generate interrupts. The INTB output is not used. To have either port generate interrupts, the INTA and INTB signals must be logically or'ed together. This is accomplished by setting the **MIRROR** Bit within the MCP23017 **IOCON** Register to "1". If using multiple port pairs as interrupt sources, configure all the INTA outputs as open-drains and enable the Pull-Up resistor (SJ3)

## Host Interrupt Selection

Solder Jumpers SJ1 – SJ7 configure which host interrupt will be associated with the board and which interrupt source will be used.

| Interrupt Configuration Solder Jumpers | | |
|---|---|---|
| Solder Jumper | Designation | Description |
| SJ1 | INT1[1] | Host Interrupt input #1, often shared with Arduino Digital I/O #3 |
| SJ2 | INT0[1] | Host Interrupt input #0, often shared with Arduino Digital I/O #2 |
| SJ3 | PU | Enable Pull-Up resistor, 4.7K to V+ |
| SJ4 | PORTG, PORTH | Port G and H Interrupt output[2] |
| SJ5 | PORTE, PORTF | Port E and F Interrupt output[2] |
| SJ6 | PORTC, PORTD | Port C and D Interrupt output[2] |
| SJ7 | PORTA, PORTB | Port A and B Interrupt output[2] |

1. Interrupt input may be shared with other system resources.
2. Ensure INTA and INTB are logically or'ed together by the MCP23017 chip software to incorporate both Ports, IOCON.6 = 1

*Try selecting a host interrupt which is not currently being used by other system resources. If interrupts must be shared, make sure all the software applications and hardware involved support interrupt sharing. To prevent excessive current draw and the possibility of erroneous operation, use only one pull-up resistor.*

# Digital I/O Software example

This example shows a basic use of the DIO64-ARD Digital I/O circuitry.

**Please refer to the MCP23017 manufacturer’s data sheet
for complete hardware and software information related to this device.**

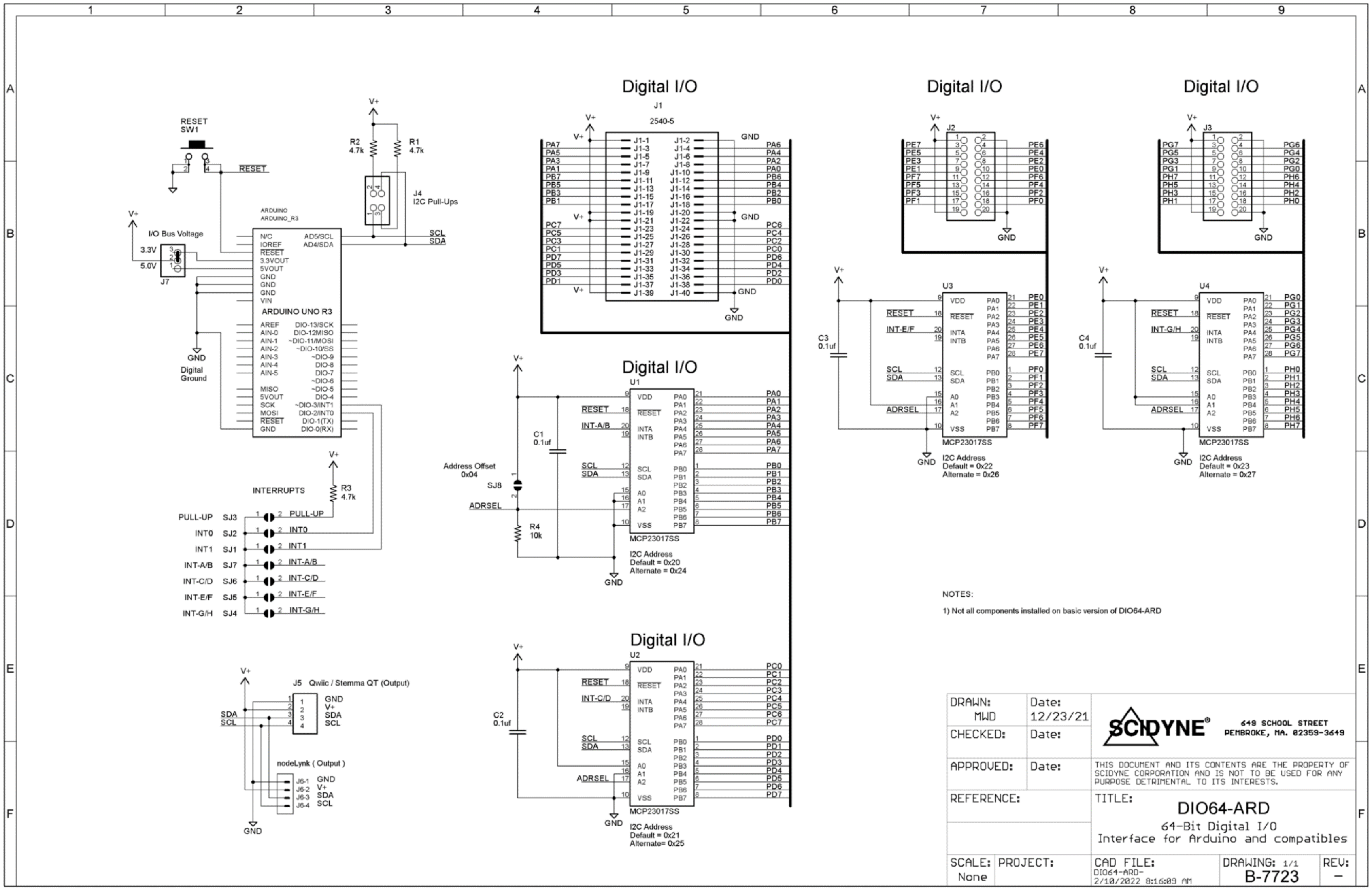
```
/* * * * * * * * * * * * * * * * * * * * * * * *
 * SCIDYNE Corporation, May 10, 2022,  Mark Durgin
 * Simple digital I/O demo program for DIO64-ARD PN #100-7728
 * - Repeatedly writes pattern of alternating 0's and 1's to Port-A of the designated MCP23017 chip
 *   Use an oscilloscope to observe results
 * - Reads state of Port-B and prints to serial monitor. Short pins to ground to cause changes
 */
#include <Wire.h>   // Needed for Arduino I2C support

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 * Program constants and defines
 */

// Define only one MCP23017 Chip Address, comment-out the other three
#define MCP23017_ADDRESS   0x20         // Chip Address for PortA & PortB
//#define MCP23017_ADDRESS   0x21       // Chip Address for PortC & PortD
//#define MCP23017_ADDRESS   0x22       // Chip Address for PortE & PortF
//#define MCP23017_ADDRESS   0x23       // Chip Address for PortG & PortH

// Define the MCP23017 registers to be used. See the MCP23017 data sheet for details
#define MCP23017_IODIRA       0x00      // MCP23017 Port-A Data Direction Register
#define MCP23017_GPIOA        0x12      // MCP23017 Port-A Data register
#define MCP23017_IODIRB       0x01      // MCP23017 Port-B Data Direction Register
#define MCP23017_GPIOB        0x13      // MCP23017 Port-B Data register
#define MCP23017_GPPUB        0x0D      // MCP23017 Port-B Pull-Up enable register

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 * setup runs once then execution goes to loop
 */
void setup()
{
  Wire.begin();                          // Setup I2C communications

  Serial.begin(9600);                   // Setup communications to IDE Monitor
    while (!Serial);

  // Setup MCP23017 Port-A as all outputs
  Wire.beginTransmission(MCP23017_ADDRESS);          // Open I2C communication with the MCP23017
  Wire.write(MCP23017_IODIRA);                       // Next byte intended for Port-A direction register
  Wire.write(0x00);                                  // Make all outputs, Bits set as 0 are outputs
  Wire.endTransmission();                            // Close I2C communications

 // Setup MCP23017 Port-B as all inputs
  Wire.beginTransmission(MCP23017_ADDRESS);          // Open I2C communication with the MCP23017
  Wire.write(MCP23017_IODIRB);                       // Next byte intended for Port-B direction register
  Wire.write(0xFF);                                  // Make all inputs, Bits set as 1 are inputs
  Wire.endTransmission();                            // Close I2C communications

  Wire.beginTransmission(MCP23017_ADDRESS);          // Open I2C communication with the MCP23017
  Wire.write(MCP23017_GPPUB);                        // Next byte intended for Port-B Pull-Up register
  Wire.write(0xFF);                                  // Bits set as 1 enable corresponding pull-up
  Wire.endTransmission();                            // Close I2C communications

}


/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 * loop runs continuously
 */
void loop()
{
  // Toggle bit pattern on Port-A
  Wire.beginTransmission(MCP23017_ADDRESS);          // Open I2C communication with the MCP23017
  Wire.write(MCP23017_GPIOA);                        // Next byte intended for Port-A data register
  Wire.write(0xAA);                                  // Make Port-A = 1 0 1 0 1 0 1 0
  Wire.endTransmission();                            // Close I2C communications

  delay(100);                                        // Wait 100ms

  Wire.beginTransmission(MCP23017_ADDRESS);          // Open I2C communication with the MCP23017
  Wire.write(MCP23017_GPIOA);                        // Next byte intended for Port-A data register
  Wire.write(0x55);                                  // Make Port-A = 0 1 0 1 0 1 0 1
  Wire.endTransmission();                            // Close I2C communications

  delay(100);                                        // Wait 100ms

  // Read the state of Port-B and display on Arduino IDE Serial Monitor
  Wire.beginTransmission(MCP23017_ADDRESS);          // Open I2C communication with the MCP23017
  Wire.write(MCP23017_GPIOB);                        // Read will be from Port-B data register
  Wire.endTransmission();

  Wire.requestFrom(MCP23017_ADDRESS, 1);             // Request 1 byte from the MCP23017
  unsigned char x = Wire.read();                     // Receive the Port-B byte and store in x
  Serial.println(x);                                 // Print the character

  delay(100);                                        // Wait 100ms

}
```

# Appendix - A Specifications

Specifications subject to change without notice

| | |
|---|---|
| **Digital I/O:** | |
| General: | Four MCP23017 chip provides 64 bi-directional I/O channels across eight 8-Bit ports |
| Output current: | ±25mA max. per output. Total limited by hosts ability to supply sufficient current. |
| Pull-Up Resistor: | 100K typical, individually software enabled on each I/O channel |
| **I$^2$C Interface:** | |
| Addressing: | Four consecutive addresses. Default: 0x20 – 0x23 Address Offset Enabled: 0x24 – 0x27 |
| Speed: | Standard (100kbps), Fast (400kbps), High-Speed (1.7Mbps) |
| Pull-Ups: | Optional Jumper selectable 4.7K Pull-Ups on SDA and SCL signals |
| **Interrupt (Optional):** | |
| Host: | One Arduino interrupt, selectable IRQ 0 or 1 |
| Pull-Up: | Optional 4.7K Pull-Up Resistor to operating voltage V+ |
| Condition: | Change-of-State or Pattern-Matching on:<br>PortA & PortB, PortC & PortD, PortE & PortF, PortG & PortH |
| **Power:** | Jumper selectable +3.3V or 5.0V. Power derived from host |
| **Physical:** | Dimensions: Standard Arduino UNO R3 footprint. Approx. 2.10” W x 3.00”L overall<br>Weight: 1.0 oz |
| **Connections:** | |
| I/O: | 40 Position IDC Ribbon Cable; PORTA, PORTB, PORTC, PORTD<br>20 Position Terminal Strip; PORTE, PORTF<br>20 Position Terminal Strip; PORTG, PORTH |
| External Expansion: | 4 Position Qwicc (optional), 4 Position nodeLynk (optional) |
| Arduino: | Stack-through connectors allows multiple shields.<br>Power: 8 Pos. x 1 Row<br>Analog: 6 Pos x 1 Row<br>Digital: 8 Pos x 1 Row & 10 Pos. x 1 Row<br>ICSP: 3 Pos x 2 Row (optional) |
| **Software:** | Uses standard Arduino I$^2$C library functions. Fully supports third-party software libraries like those from Adafruit and SparkFun. |
| **Environmental:** | |
| Operation: | -25°C to 65°C (Standard) Non-condensing relative humidity: 5% to 95% |
| Compliance: | RoHS |
| **Product Origin:** | Designed, Engineered, and Assembled in U.S.A. by SCIDYNE® Corporation using domestic and foreign components. |
| **HTS Code:** | 854231 |

# Appendix - B  Schematic Diagram



## User Notes