PIXIE BOARD
LABVIEW SOFTWARE USER MANUAL

AEL Microsystems Ltd
Malvern
UK

Acknowledgements:

AEL Microsystems Ltd acknowledges the trademarks of other organisations for their respective products and services mentioned in this document.

This contact information is subject to change, for the latest details go to www.aelmicro.com

Web: https://www.aelmicro.com
Email: pixie.support@aelmicro.com

1. Contents

2. Introduction

2.1 Caution

This board is designed using modern CMOS devices, observe standard anti-static procedures when handling this board otherwise permanent damage may result.

You have been warned !

2.2 Forward note

Thank you for choosing one of the **PI** e**X**pansion **I**ndustrial **E**lectronic boards, **"PIXIE"**.

The range of **PIXIE** boards has been developed to allow you to expand the hardware functionality of your Raspberry Pi, by adding one or more **PIXIE** boards gives you a wider range of interface solutions. The **PIXIE** range of boards has been developed to allow for the Raspberry Pi to be used in harsher industrial and real-world environments.

The key objective of a PIXIE board is:

- Provide an expansion board to allow the use of a Raspberry Pi in industrial environments.
- Allow for more than one expansion board to be stacked onto an existing Raspberry Pi unlike a HAT.
- 16 boards can be stacked and given a unique logical address using the board selector switches.
- Provides a low-cost industrial control solution.
- Standard board profile which is the same as the Raspberry Pi.
- Optional enclosure to allow mounting direct to industrial DIN rail.
- Fully software configurable, i.e. no links to set.
- Can use either SPI devices 0 or 1.
- Supported is provided for National Instruments LabVIEW.
- Comes with fully supported **PIXIE** software API and libraries, for 'C', 'C++' and Python

The **PIXIE** boards have not only been developed for use solely with the Raspberry Pi but can easily be interfaced to other microcontrollers and CPU modules allowing your project to be based on alternative platforms and operating systems.

All **PIXIE** boards use a standard size board and are connected to the Raspberry Pi board using the 40-way IDC connector.

Multiple **PIXIE** boards can be stacked on to the Raspberry Pi and once assembled can be configured using the **PIXIE** board configuration and update utility eliminating the need to dismantle the board stack to change the settings.

3. Common concept

3.1 Control overview

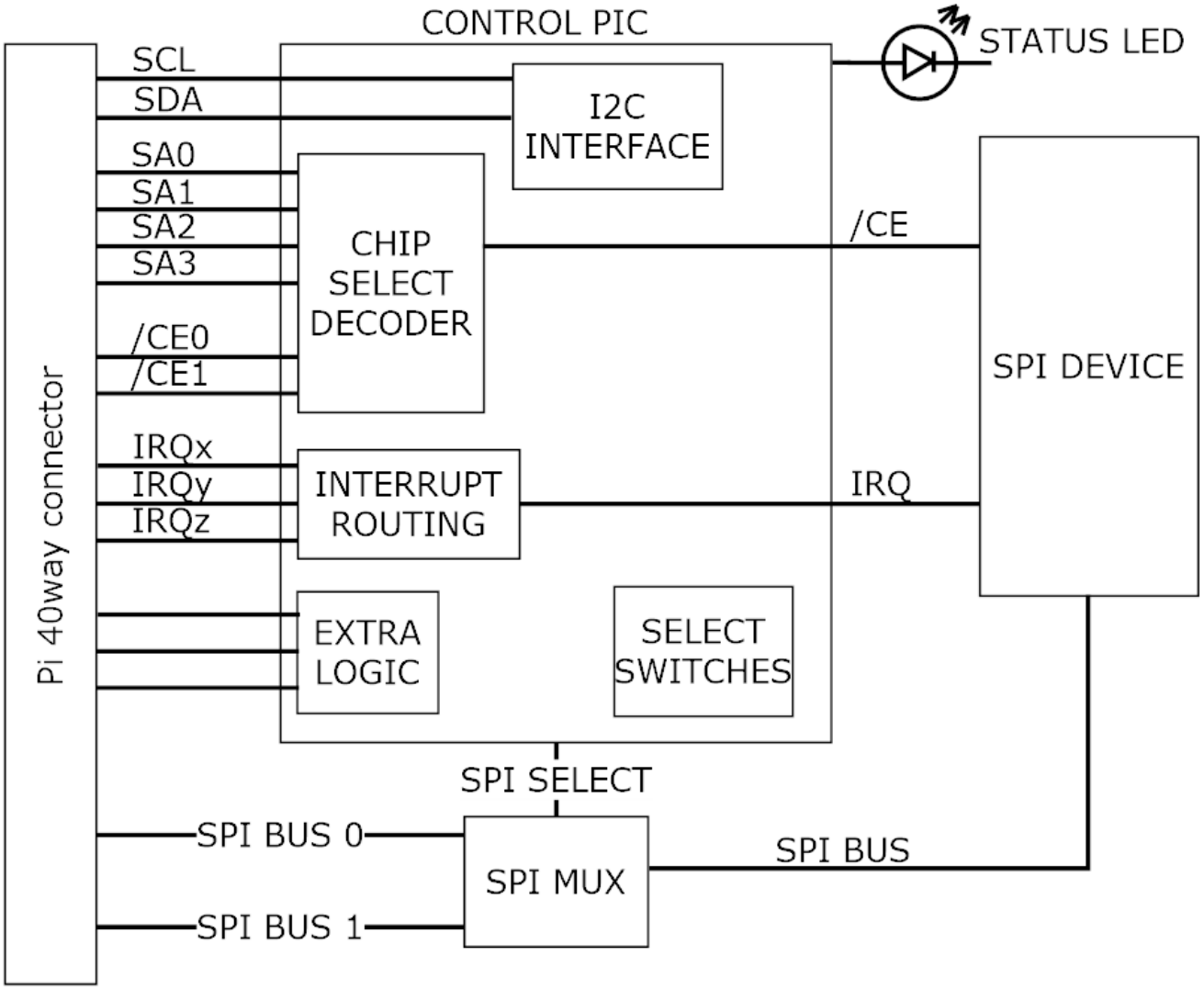
This shows the control overview of the **PIXIE** board.

The select switches give each **PIXIE** board a unique identity.

The Raspberry Pi communicates using the I2C bus to configure the **PIXIE** board.

The sub address (SAx) signals, interrupt signals (IRQx) and extra logic signals are connected to the GPIO pins of the Raspberry Pi.

Either SPI0 or SPI1 bus is routed to the board devices



The design goal of the **PIXIE** board range is to provide the user with a board that has a common footprint, uses no configuration links, and once assembled into a stack with the Raspberry Pi, can be configured and used without the need to make any further physical changes except for wiring in the connectors. All boards use 3.5mm two-part pluggable terminal blocks which in the event of a board change or other upgrade, can be simply unplugged without the need for a screwdriver.

IT IS NOT A HAT
The boards are not HAT's, their biggest difference is that you can stack up to 16 onto the Raspberry Pi and they all use the SPI busses for maximum software access, nor do they use the HAT configuration memory.

3.2 More SPI devices

This concept is achieved by the use of some of the GPIO signals to provide additional address signals used for decoding the SPI chip selects found on the 40-way connector. You can have up to 4 additional SPI address signals per SPI bus, these are qualified by the onboard hardware decoding to give you up to 16 possible decode addresses for each SPI bus chip select. So, for SPI0 it has 2 chip selects, that is 32 possible devices, for SPI1 it has 3 chip selects, that is 48 possible devices, 80 in total which is more than most needs.
Each board can be configured to use as many of the address signals as it requires as well as which chip select the board will use.
To facilitate this sub address system requires changes to the SPI device driver and rebuild the kernel or use the precompiled SPI device driver and install it on the Raspberry Pi to replace the current one.

3.3 Configurable

Each board is software configurable from the Raspberry Pi using a simple command line application called "**PixieBoard**". Each board is given a unique identity which is set by the small piano key switch allowing each board to be numbered 0 to 15. The board is configured over the I2C bus using a base address of 0x10 plus the value of the piano switch giving a range of unique I2C addresses from 0x10 to 0x1F. Each board can then be accessed individually and configured as required.
All the configuration values for each board are stored in EEPROM memory on the board so once it has been configured it does not have to be reloaded whenever the board is power cycled.

The key configurable items of each board are:

- Which SPI to use, SPI0 or SPI1
- Which chip SPI selects to use, CE0, CE1 or CE2
- Which SPI sub address signals to use and the sub address value to decode.
- Which GPIO will receive an interrupt if required from the board.
- Additional board specific settings.

3.4 Stackable

Each board can be stacked on top of each other and the Raspberry Pi using 17mm spacers or enclosed in one of the plastic housings which allows for the use in a more robust environment as well as mounting to a standard industrial DIN rail.
As previously mentioned up to 16 boards can be stacked together.

3.5 Updatable

All boards make use of a small microcontroller to interface to the Raspberry Pi over the I2C bus, and provide the real time hardware decoding logic and other board support functionality. If at any point new firmware is made available, the "**PixieBoard**" application can be used to update the boards firmware without the need to dismantle the board from the stack or use any external programme

4. Installing LabVIEW support for PIXIE

This section describes the installation of the support libraries for PIXIE boards to run with LabVIEW on a Raspberry Pi.

4.1 Tools required

LabVIEW community, academic or professional, version 2020 or greater depending upon the licence you hold.

Install LabVIEW support on your Raspberry Pi by following the installation instructions then test that you can run a LabVIEW application, all the details are provided on the National Instruments website.

On the Raspberry Pi
Download the pre-built PIXIE support library **pixie-lv-library-x.y.tgz** found at
aelmicro.com > Support > Pixie Boards > Software > Pixie LabVIEW support library onto your Raspberry Pi.

Open a command line window and enter the following:

```
cd /srv/chroot/labview/usr/lib
sudo tar xzf  ~/Downloads/pixie-lv-library-x.y.tgz
cd ~
```

Using the Raspberry Pi file manager verify for the presents of the shared library
/srv/chroot/labview/usr/lib/libpixielv.so

On your Windows host
A VI package manager PIXIE support package is available and adds the PIXIE support functions to the LabVIEW palettes.

Obviously, you need LabVIEW installed on your PC and also the VI Package manager which will install the support package onto LabVIEW.

Download the LabVIEW support package **ael_microsystems_limited_lib_pixie_support-x.x.x.y.vip** found at **aelmicro.com > Support > Pixie Boards > Software > Pixie LabVIEW support VIP** onto your PC.

Open the VI package manager and locate and select the downloaded ***.vip** file.

Install the package onto LabVIEW.

The installation will add the PIXIE support VI's and controls to the palette group "**PIXIE support**" and some examples can be located using the LabVIEW **Help > Find examples**, select directory structure and **AEL Microsystems Limited > PIXIE Support**.

Load up the **Pixie Example.lvproj** and set the IP address in its properties for your Raspberry Pi.
Running the **Pixie Get Pi Info.vi** is a good place to begin as it will do a simple sanity check of the connection and return information about your Raspberry Pi.

5. Compiling LabVIEW support for PIXIE

This section describes the building of the **libpixielv.so** shared library.

Due to the different settings of the GNU compiler it is not possible to recompile the shared library using the tools on the Raspberry Pi, it has to be done using the cross compiler supplied by National Instruments. Following is the procedure to do this using a Windows PC Host.

5.1 Tools required on the PC host

- National Instruments GNU cross compiler for ARM
- 7-Zip utility.
- GNU Make utility.

Download the cross-compiler tools for Windows from the National Instruments website, this can normally be found by doing a search "National Instruments Linux Arm compiler".
The relevant page should be something along the lines of "GNU C & C++ Compiler Tools for ARMv7"

Get and install 7-Zip
Find, download, and install the "7-zip" utility which is used to open the archive.

Get and install cross compiler
Select the "Windows host" and the latest version and download the *.tar.xz file.
Open the *.tar.xz file and then double click the *.tar archive shown within, this will open into a temporary folder and may give an error at the end, just ignore this for now.

From the root folder double click to expand the folder and 5 items are displayed, a **sysroot** folder and 4 support files.

Using Windows Explorer create a folder in the root of the C drive called **C:\NIxcc**
If you use a different folder name, then you will need to edit the **setup.bat** file used later on to point at the correct folder.

Extract all files and folders into it.

Get and install GNU Make
Find and download a copy of GNU make for windows.
MinGW is a great place to start and from the installation package manager you just need to install **mingw32-make-bin**, the files appear in **C:\MinGW\bin** as **mingw32-make.exe**
When running the setup.bat later on it will set up a local symbolic link **make.exe** to the **mingw32-make.exe** making it easier to just use "make".

5.2 Source installation

Download the source code **pixie-lv-source-x.y.tgz** found at
aelmicro.com > Support > Pixie Boards > Software > Pixie LabVIEW support source onto your PC.

Open the *.tgz file and then double click the *.tar archive shown within, this will open into a temporary folder.

Using Windows Explorer create a project folder of your choice to extract all the folders and files into.

Extract all files and folders into it.

5.3 Source compilation

Open a command window and change to the folder in which you installed the source file set into.

If you installed the Mingw32-make and the cross compiler in different folders to those described earlier then you will need to edit the setup.bat file to set the file paths to the correct ones, below are the defaults.

```
set XCC_ROOT=C:\Nixcc
set MINGW_PATH=C:\MinGW
```

Run “setup.bat”, this batch file will also install a symbolic link called **make.exe** pointing to the **mingw32-make.exe** allowing the use of “make” locally for building.

To clean the last build

make CONFIG=Release clean

To build the library

make CONFIG=Release

The output library is found in **Release\libpixielv.so**

Copy this library to the Raspberry Pi using something like **SmarTTY** from **sysprogs.com**, then install the transferred file into the **/srv/chroot/labview/usr/lib** overwriting the existing one.

Reboot the Raspberry PI to ensure the latest library is used.

6. Warranty conditions

All fully assembled & tested products of AEL Microsystems Ltd are guaranteed for one year from the date of shipment against defects in materials & workmanship and perform in accordance with applicable specifications. AEL Microsystems Ltd warrants that the application support SOFTWARE will perform substantially with the accompanying written materials for a period of ninety (90) days from the date of receipt.

This warranty does not extend to products which have been altered or repaired by persons other than persons authorised by AEL Microsystems Ltd, or to products that have been subjected to misuse, abuse, neglect, improper installation or application, accident, disaster, or modification not approved by written instructions from AEL Microsystems Ltd.

Final determination of the suitability of this product for the use contemplated by the buyer is the sole responsibility of the buyer and AEL Microsystems Ltd shall not be responsible for its suitability and assumes no liability arising out of the use or application of the device described herein.

In the event that this product fails to operate as warranted, the buyer shall obtain a return number from AEL Microsystems Ltd and forward the product in suitable packaging with a detailed failure report to AEL Microsystems Ltd, the cost of transportation being the responsibly of the buyer. The returned product will be repaired or replaced at the discretion of AEL Microsystems Ltd.

While every effort is made to repair or replace any item as quickly as possible, no guarantees can be made for the time taken, & AEL Microsystems Ltd cannot be held responsible for any loss or inconvenience caused.

7. Notes