GUIDE TO OPERATION OF ACOUSTIC DATA COLLECTION SYSTEM (AQFI-1301) FOR SHALLOW WATERS

Development and improvement of acoustic equipments and systems for shallow areas

Research Institute for Humanity and Nature (RIHN)
GUIDE TO OPERATION OF ACOUSTIC DATA COLLECTION SYSTEM (AQFI-1301) FOR SHALLOW WATERS

Development and improvement of acoustic equipments and systems for shallow areas

Researchers:

Yoshinori Miyamoto\(^1\), Keiichi Uchida\(^1\),
Yuttana Theparoonrat\(^2\), Monton Anongponyoskun\(^3\),
Kritsada Thongsila\(^4\), Yap Minlee\(^1\), Toyoki Sasakura\(^5\),
Kohei Hasegawa\(^6\)

\(^1\) Tokyo University of Marine Science and Technology, Japan
\(^2\) Southeast Asian Fisheries Development Center, TD, Thailand
\(^3\) Faculty of Fisheries, Kasetsart University, Thailand
\(^4\) Eastern Marine Fisheries Research and Development Center, DOF, Thailand
\(^5\) Fusion Inc., Japan
\(^6\) National Research Institute of Fisheries Engineering,
Japan Fisheries Research and Education Agency, Japan
Marine resources are important protein sources for the growing human population, but these resources can be difficult to manage because of factors such as their inaccessibility and variability. Acoustic survey technology can help to overcome some of these difficulties. In particular, the development of acoustic equipment and analytical systems to process the acoustic data have been very beneficial for marine stock assessments, especially in deep areas such as the high seas. Here, I would like to mention other benefits of applying new acoustic technologies towards natural resource management in coastal areas of tropical zones.

Although coastal areas are relatively small in relation to the area of the open ocean, they contain high amounts of biodiversity and primary productivity and thus are very important for resource sustainability. In general, many small-scale fisheries are conducted in coastal areas, and these fisheries play important roles as income and protein sources for local people. Unfortunately, collecting data on small-scale fisheries can be quite difficult, in part, because much of the activity is family-based, the products are traded during face-to-face interactions. Therefore, collaboration is needed among the local fishers, researchers and governmental officers to collect reliable fisheries statistics data toward sustainable fisheries developments. In this regard, it is important to find better ways to involve the local fishers in stock assessments and fisheries management decisions.

A fishery management typically begins with a stock assessment based on statistical data, and then, regulate fishing activities up to target yields e.g. MSY etc. Authorities responsible for fisheries management then enforce the corresponding regulations. This approach seems quite rational, but it does not do enough to facilitate the participation of local fishers. There are typically no incentives for local fishers to collaborate with the managing authorities. The establishment of a system in which local fishers are given high incentives to participate in resource management would be very beneficial for coastal fisheries.

A community-based set-net fishery was installed along a beach in Rayong, Thailand, in 2003, and catch data from this fishery during 2003 to 2013 were well collected by fishermen. This success shows us how best to involve local fishers in data collection efforts. Government officials and researchers provided some initial technical and economic support for the set-net installation, and local fisheries groups were directly involved in product sales and community-based activities such as data collection. Catch data and selling data were shared among the members. Moreover, the set-net could act as a nursery for larval and juvenile fish, and as a physical barrier against illegal fishing by large vessels in the coastal zone. The members of the set-net fisheries groups demonstrated concern about the fish inside the set-net. Acoustic technology was employed to give them information about the actual number of fish in the net. If the technology actually installed to set-net, incentive of fishermen to collaborate with researchers will also be enhanced because this technology improved the efficiency of fisheries management efforts.
If local fisher groups are able to use acoustic technology to assess and monitor the fishery resources, they are better able to understand the status of the fisheries and they can utilize the resources effectively. As the depth of the coastal area is shallow in many locations, traditional acoustic surveys for stock assessments are not always a suitable option. However, some acoustic survey systems and equipment have been invented recently those are applicable to shallow waters. Even though these shallow-water systems do not have high resolution capabilities like the systems used in high sea areas, they can provide useful information and help communities visualize the fisheries resource situation in the underwater environment. Such information was found to be important for unifying the local fishers who operate in the same fishing grounds.

Recently, area-capability cycle was proposed as a model for balanced development in rural areas. The area-capability cycle consists of finding new resources, effectively using them, developing a community of users, enhancing user capability, cultivating interest in the health of the ecosystem, understanding the importance of caring for the environment, promoting activities to care for natural habitats and primary production of resources, and fostering pride and hope in users in regards to their use and care of ecosystem services. New technology can help find new resource; new technology can facilitate effective utilization of the resource. And new technology can promote collaboration of many stakeholders. What I would like to say is the development of acoustic technology can be a key for new coastal development model.

Satoshi ISHIKAWA
Research Institute for Humanity and Nature (RIHN)
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreword</td>
<td>i</td>
</tr>
<tr>
<td>Contents</td>
<td>iii</td>
</tr>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Acoustic Data Collection System for Shallow Waters</td>
<td>2</td>
</tr>
<tr>
<td>How to Connect Echo Sounder to Data Collection System</td>
<td>4</td>
</tr>
<tr>
<td>Setting Up of Acoustic Data Collection System</td>
<td>10</td>
</tr>
<tr>
<td>System Software Installation and Configuration</td>
<td>13</td>
</tr>
<tr>
<td>A. Pico Scope 6</td>
<td>13</td>
</tr>
<tr>
<td>B. Aqua Sound</td>
<td>18</td>
</tr>
<tr>
<td>C. Hyper Terminal</td>
<td>19</td>
</tr>
<tr>
<td>FURUNO GP-1670F Operational Procedure</td>
<td>24</td>
</tr>
<tr>
<td>Data Collection Operation Procedure</td>
<td>27</td>
</tr>
<tr>
<td>Echo Sounder Data Analysis Procedure</td>
<td>30</td>
</tr>
<tr>
<td>GPS Data Analysis Procedure</td>
<td>43</td>
</tr>
<tr>
<td>Annex I: Source Code of Program “Fish Finder” Version 3.2 in Visual C</td>
<td>52</td>
</tr>
</tbody>
</table>

※ About trademarks: FURUNO, RotoKey are trademarks or registered trademarks of Furuno Electric Co., Ltd.
INTRODUCTION

Research Institute for Humanity and Nature (RIHN) program,

**Component 5 Activities:**
Activity1: Developing the new data collection equipments and analysis systems of acoustic survey at coastal areas
Activity2: Field test of the developed equipments and system
Activity3: On-site-Training on new acoustic survey equipments and systems
Activity4: Publication of the research protocol guidelines for acoustic survey in coastal areas

**Development and improvement of acoustic equipments and systems for shallow area**

This study was carried out as part of the RIHN project on “Coastal area-capability enhancement in Southeast Asia”. This project was aimed at investigating the linkage between livelihoods and ecosystem health in the Southeast Asian coastal areas to fully understand its complexity and consequent vulnerability, particularly from the human-related viewpoints through collaborative holistic researches with local peoples. The acoustic data collection system including equipment modification was conducted for coastal area survey around the set-net fishing ground in Rayong Province, Thailand. An analytical methodology was also developed and used as a tool for acoustic survey methodology and education for young scientists. Since, the depth of the target research area is about 15 meters, the searching range is too narrow for using scientific echo sounder available in the market, the project therefore modified the acoustic device which is composed of GPS Plotter Fish-Finder (FURUNO GP1670F), Interface box and personal computer (PC), in order that this could be used as acoustic data collection equipment in coastal areas.

**Objectives**

1) To develop new acoustic data collection system for shallow waters,
2) To study the fisheries resources distribution around set-net fishing ground in Rayong Province, Thailand,
3) To estimate the amount of fisheries resources using acoustic data and the fish catch data of set-net fishing operation, and
4) To conduct human resource development activities on the new acoustic survey equipment and system through on-site training and publication of guidelines for acoustic survey in coastal areas.
Development of new data collection equipments and analysis systems for acoustic survey in coastal areas.

Development of acoustic data collection system and modification of equipment were carried out for the coastal area survey in Rayong Province, Thailand. The hardware and software system for data collection were developed at Tokyo University of Marine Science and Technology, Tokyo Japan. The first testing session of the hydro-acoustic equipments and systems for shallow areas were conducted at Tateyama Bay, Chiba Prefectures, Japan in October 2012.

**Modification of GPS and echo sounder system for hydro-acoustic data collection**

The hydro-acoustic system for data collection was modified by using any kind of echo sounder and GPS receiver. Some devices have echo sounder and GPS plotter (e.g. FURUNO GP-1670F), but some devices are independent of an echo sounder and the GPS receiver.

**Required specifications of echo sounder**

The echo sounder should operate on 50 kHz. The data collection system was designed to record the echo sounder signal of 50 kHz only.

**Required specifications of GPS receiver**

The GPS receiver provides highly accurate position, courses and speed information. The GPS receiver should have these data output. Normally, these data are in NMEA0183 format and output in RS-232C or USB terminal are used for other instruments.

![Echo sounder data collection system block diagram.](image)

In this study, the hydro-acoustic system for data collection was modified by using FURUNO GPS Plotter model GP-1670F. The GP-1670F is equipped with GPS receiver and chart plotter system. The machine is also equipped with echo sounder operating simultaneously on 50 kHz and 200 kHz. The GP-1670F provides a total integrated GPS receiver, color video plotter and color fish finder. The built-in GPS receiver provides highly accurate position, courses and speed information. The fish finder presents vivid underwater images on a high quality LCD.
The compact display unit and antenna unit permit could be installed even where space is limited. Specifications of the equipment are shown as follows,

Main features of GP-1670F;

- Bright 5.7 inch (GP-1670F) color LCD with brilliance control.
- Excellent viewing angles, even when wearing sunglasses.
- Internal GPS receiver provides high accurate position information (GPS, within 2.5 m, SBAS, within 2 m).
- Customizable analog and digital displays show wind angle and speed, engine condition (speed, temperature, oil pressure, etc.), etc.
- Large internal memory stores 30,000 track points, 30,000 points, 1,000 routes (500 waypoints/route).
- SD card slot accepts SD and SDHC cards for external storage of data and settings.
- Full range of alarms; Arrival, Anchor Watch, Cross-track Error, Speed, Depth, Temperature, Fish Alarm, Bottom Alarm, etc.
- Man overboard (MOB) feature records latitude and longitude coordinates at the time of MOB.
- CAN bus interface for the connection of GPS Receiver, Weather Station, FI-50 (instrument series), Satellite Compass, etc.
- Accepts NMEA0183 input with optional NMEA data converter.
- Internal GPS antenna available.
- C-Map 4D charts available.

![Fig 2](image.png)

Figure 2. FURUNO GP-1670F display and control panel.

※ This figure is quoted from “FURUNO GPS PLOTTER/SOUNDER GP1670F, GP1870F OPERTOR’S MANUAL” of product GP-1670F of Furuno Electric Co., Ltd.
**HOW TO CONNECT ECHO SOUNDER TO DATA COLLECTION SYSTEM**

There are multiple signal lines on the cable connecting the main unit and the transducer. Among them, it is necessary to find the signal line used for transmission/reception. The number of pins and assignments varies from manufacturer to manufacturer. In many cases, it is described in the manual of each device. After checking the signal line, cut the cable halfway and connect it to terminal 8 (Transducer side) shown in Figure 4. Other signal lines are connected to the same pin number as Transducer side and GP1670F side. If necessary, replace the connector both of Transducer side and GP1670F XDR PORT.

Equipment modification was performed by importing the echo sounder transmitting and receiving signal from transducer cable to the newly designed Echo Sounder Data Collection System (AQFI-1301). The Echo Sounder Data Collection System (AQFI-1301) is composed of Pre-Amplifier and Band Pass Filter, Interface unit, Analog to Digital Converter, and PC computer system. The GPS position data is transferred from GP 1670F to PC Computer passing through NMEA to USB port. The GPS position data are recorded to the PC using Windows Hyper-terminal program. The data collection system is designed to record the echo sounder signal of 50 kHz. only.
Figure 4. Circuit diagram of Survey Echo Sounder Data Collection System (AQFI-1301).
Figure 5. Frequency filter circuit at 50 kHz.
Figure 6. Pre-Amplifier circuit at 50 kHz.
The return echo signal picks-up from the transducer cable is fed into Pre-Amplifier circuit for signal filtering and amplifying. Analog echo signal from Pre-Amplifier are put into Analog – Digital Converter, Pico 4224, for transforming into echo signal digital data. Personal Computer receives and records digital echo signal by AQUA SOUND software program of “FishFinder, Version 3.2” for Microsoft Windows 7. Digital data from echo sounder are sent to PC through USB port.

**Figure 7.** Interface circuit using IC 2122.

**Figure 8.** Analog to Digital Converter, PicoScope 4224.
How to connect GPS receiver to PC

Ship position information (GPS) from echo sounder (GP1670F) is present on the CAN bus equipment port. CAN bus is a communication protocol that shares multiple data and signals through a single backbone cable. By simply connecting any CAN bus devices into the backbone cable, the network onboard can be expanded. All CAN bus devices can be incorporated into the NMEA2000 network.

The GPS receiver should have USB or RS232C data output port. If it is a USB output, connect it directly to the PC. If it is RS232C, connect it to the PC using the conversion cable (Figure 10).

The GPS information from echo sounder is transferred to the PC by applying the program “HyperTerminal” through NMEA Data Converter (IF-NMEA2K2), USB-to-Serial (RS232) Converter, and PC USB port.

![Figure 9. NMEA Data Converter (IF-NMEA2K2).](image)

![Figure 10. USB-to-Serial (RS232) Converter.](image)
SETTING UP OF ACOUSTIC DATA COLLECTION SYSTEM

The acoustic data collection system hardware comprises of three main components such as Echo sounder (GP-1670F), Data Collection System (AQFI-1301), and Data Collecting Unit (PC). The set up of the actual acoustic data collection system hardware is shown in Figure 11.

![Acoustic data collection system hardware connection layout](image)

**Figure 11.** Acoustic data collection system hardware connection layout.

Setting up the acoustic data collection system hardware could be carried out following the procedures below:

1. **Power cable connection**
   Connect the power cable to the power connector of FURUNO GP-1670F and Echo Sounder Data Collection System (AQFI-1301). Connect the leads to the battery (12 or 24 VDC): red to plus (+) terminal and black to minus (-) terminal. Make sure that the shield is safe by connecting it to a ground which is the ship’s ground.
   Note: Since the fuse (3A) is not waterproof, wrap the fuse holder with vinyl tape to keep water out from the fuse holder.

![Power cable connection](image)

**Figure 12.** Power cable connection.
2. Transducer connection
Connect the transducer cable to the XDR port of GP-1670F and ES port of AQFI-1301. Connect transducer cable, (520-5PSD) to TD port of AQFI-1301.

Figure 13. Transducer connection.

3. CAN bus (GPS Information) connection
Connect FURUNO CAN bus cable of NMEA Data Converter (IF-NMEA2K2) to CAN bus port of GP-1670F. Connect RS232 Serial port (9-pins) of NMEA Data Converter (IF-NMEA2K2) to USB-to-Serial (RS232) Converter. Connect USB of USB-to-Serial (RS232) Converter to computer.

Figure 14. CAN bus (GPS Information) connection.

4. Data output connection
Connect data cable USB type B male to Data Out of Echo Sounder Data Collection System (AQFI-1301) and USB type male A to Computer USB port.

Figure 15. Data output connection.
Mount a Transducer

The thru-hull mount transducer provides the best performance of all, since the transducer protrudes from the hull and the effect of air bubbles and turbulence near the hull skin is reduced. If the boat has a keel, the transducer should be at least 30 cm away from it.

The performance of fish finder is directly related to the mounting location of the transducer, especially for high-speed cruising. Thus, the installation should be planned in advance, keeping in mind the length of the transducer cable and the following factors:

- Air bubbles and turbulence caused by movement of the boat seriously degrade the sounding capability of the transducer. The transducer should, therefore, be located in a position where water flow is the smoothest. Noise from the propellers also adversely affects performance and the transducer should not be mounted nearby. The lifting strakes are notorious for creating acoustic noise, and these must be avoided by keeping the transducer inboard.
- The transducer must always remain submerged, even when the boat is rolling, pitching or planning (gliding) across water at high speed.
- A practical choice would be to place the transducer somewhere between 1/3 and ½ of the boat’s length from the stern. In the hull, it is generally practical to install or locate the transducer rather far from the boat stern, so that is always in water regardless of the planning attitude of the boat.
SYSTEM SOFTWARE INSTALLATION AND CONFIGURATION

A. **Pico Scope 6**

1) Insert Pico Technology system disk to computer

2) Select “pico” Application, Enter

<table>
<thead>
<tr>
<th>Name</th>
<th>Date modified</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>documents</td>
<td>29/8/2554 16:15</td>
<td>File folder</td>
<td></td>
</tr>
<tr>
<td>tscommand</td>
<td>29/8/2554 16:15</td>
<td>File folder</td>
<td></td>
</tr>
<tr>
<td>software</td>
<td>29/8/2554 16:15</td>
<td>File folder</td>
<td></td>
</tr>
<tr>
<td>utilities</td>
<td>29/8/2554 16:15</td>
<td>File folder</td>
<td></td>
</tr>
<tr>
<td>autorun</td>
<td>24/9/2552 18:15</td>
<td>Setup Information</td>
<td>1 KB</td>
</tr>
<tr>
<td>pico</td>
<td>27/5/2554 17:17</td>
<td>Application</td>
<td>3,923 KB</td>
</tr>
<tr>
<td>PicoAutomotiveCatalogue</td>
<td>31/8/2553 16:10</td>
<td>Adobe Acrobat D...</td>
<td>3,713 KB</td>
</tr>
<tr>
<td>PicoCatalogue</td>
<td>6/6/2554 16:43</td>
<td>Adobe Acrobat D...</td>
<td>3,331 KB</td>
</tr>
<tr>
<td>readme</td>
<td>29/8/2554 13:40</td>
<td>Text Document</td>
<td>4 KB</td>
</tr>
</tbody>
</table>

3) Select “Run Pico.exe” from AutoPlay, Enter

4) Select language menu as “English”, Enter
5) Select “USB oscilloscope”, Enter

6) Select “Install Software”, Enter

7) At Install Software, select “Install Software”, Enter
8) Select “Install PicoScope and PicoLog”, Enter

9) Select “Install PicoScope 6”, Enter

10) Select the language for the installation from the choices given such as “English (United States)”, OK
11) At Welcome to the Install Shield Wizard for PicoScope 6, select “Next >”, Enter

![Welcome to the Install Shield Wizard for PicoScope 6]

12) At License Agreement window, select “I accept the terms in the agreement”, Next >, Enter.

![License Agreement]

13) At Ready to Install the Program, select “Install”, Enter

![Ready to Install the Program]
14) During Installing PicoScope 6, please wait while the installShield Wizard installs PicoScope 6 is finished,

15) On windows Security asking for installing this device software, select “Always trust software from “Pico Technology Ltd.”, “Install”, Enter

16) At InstallShield Wizard Completed windows, select “Finish”, Enter
17) Select “Quit” on Pico Technology Install Software windows

![Install Software Window](image)

18) After completing the PicoScope 6 installation, the “PicoScope 6” icon will show on the computer display screen

![PicoScope 6 Icon](image)

B. **Aqua Sound**

1) Insert FishFinder Version 3.6 system disk of AquaSound into computer

![FishFinder V3.6 Disk](image)
2) Copy V3b6_FishFinder application to drive c: V3b6_FishFinder application icon will appear on drive c: screen.

C. Hyper Terminal

1) Load Hyper Terminal software into the computer. Select “HyperTrm” Application, Enter

![Hyper Terminal software](image)

2) Hyper Terminal program start-up will appear on display as follows:
3) Connection Description windows will appear for information selection

4) Enter Name of connection, (For example: “GPS – Hyper Terminal”)
Select type of icon for current connection

5) At “Connect To” windows, select “Connect using:” by selecting from available COM port. Available COM port can be checked by following steps: Control Panel > Hardware and Sound > Device and Printer > Device Manager > Port (COM & LPT).
6) At “Port Setting” windows, choose “Bits per second:” as 4800.

7) At “Port Setting” windows, choose “Data bits:” as 8

8) At “Port Setting” windows, select “Parity:” as None
9) At “Port Setting” windows, select “Stop bits:” as 1

10) At “Port Setting” windows, select “Flow control:” as Xon/Xoff

11) After completing Port Setting, enter OK.
12) The Hyper Terminal program will start-up with GPS – Hyper Terminal module as already set up.
FURUNO GP-1670F OPERATIONAL PROCEDURE*

1) Turn the Power On, FURUNO GP-1670F

Press \( \text{\textbullet} \) to turn the power ON.

※ This figure is quoted from “FURUNO GPS PLOTTER/SOUNDER GP1670F, GP1870F OPERATOR’S MANUAL” of product GP-1670F of Furuno Electric Co., Ltd.

2) Menu Operation

This section shows how to operate the menu. There are eight menus, [General], [Map], [Plotter], [Alarms], [System], [Fish Finder], [Instruments] and [Interface].

Long-push of the ESC/MENU key will show the main menu. Select “FISH FINDER”

3) Select “BOTTOM RANGE SHIFT AREA” for RANGE selection
[RANGES]: The default range settings are suitable for most applications. However, the ranges can be customized to suit one’s needs, i.e. [RANGE 1] - [RANGE 8]. Set the ranges in descending order. Make sure that each range is lower than its preceding range.

<table>
<thead>
<tr>
<th>Range</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANGE 1</td>
<td>15 M</td>
</tr>
<tr>
<td>RANGE 2</td>
<td>20 M</td>
</tr>
<tr>
<td>RANGE 3</td>
<td>60 M</td>
</tr>
<tr>
<td>RANGE 4</td>
<td>100 M</td>
</tr>
<tr>
<td>RANGE 5</td>
<td>120 M</td>
</tr>
<tr>
<td>RANGE 6</td>
<td>200 M</td>
</tr>
<tr>
<td>RANGE 7</td>
<td>300 M</td>
</tr>
<tr>
<td>RANGE 8</td>
<td>500 M</td>
</tr>
<tr>
<td>ZOOM RANGE</td>
<td>30 M</td>
</tr>
<tr>
<td>BOTTOM LOCK RANGE</td>
<td>30 M</td>
</tr>
</tbody>
</table>

Select a display range
Range can be selected automatically or manually. Open the RotoKey menu then select [Auto Range] and [Auto] or [Manual].
Select manual operation, the [RANGE] window, which appears on the right. Turn the key clockwise to increase the range and counterclockwise to decrease the range. Select the Range at 20 m.

4) Adjust the Gain
The gain controls how echoes of different strengths are displayed. Gain is automatically adjusted; however, the gain can be fine tuned accordingly to meet local characteristics, etc. Set the gain to show a slight amount of noise on the screen. Increase the gain for greater depths and lower the gain for shallow waters.

To adjust the gain, open the RotoKey menu then select [Gain 50k] or [Gain 200k] followed by the soft control labeled with the frequency that is desired to be adjusted, then the corresponding adjustment window appears. Rotate the key clockwise to increase the gain, counterclockwise to decrease it.
**How to adjust the gain**

Since the gain controls how echoes of different strengths are displayed, set the gain to show a slight amount of noise on the screen. Increase the gain for greater depths and lower the gain for shallow waters.

Gain too high          Gain proper          Gain too low

---

**CAUTION**

Adjust the gain correctly. Incorrect adjustment can lead to a dangerous situation especially if the boat is steered according to the depth indication.

5) **Home Screen (Display Selection)**

Select a display from the home screen which has eight displays. Press the HOME/CTRL key to show the home screen. Operate the CursorPad or rotate the RotoKey™ to select a display. The current selection is circumscribed by a red rectangle. Press the RotoKey™ or ENT key to confirm your selection.

Select a display for the right half then push RotoKey™. For example, select the fish finder display. Control returns to the home screen. The result of your selection appears on the home screen.

**Plotter, fish finder display**
DATA COLLECTION OPERATION PROCEDURE

1) Turn on FURUNO GP-1670F
   - Select the Range at 20 m.

2) Turn on Computer with Windows 7

3) Start PicoScope 6

4) Start FishFinder V.3.6

When FishFinder program is running, a window on monitor screen will display an echogram which is indicated by water depth range and echo color scale.
5) Start Hyper Terminal: (GPS – Hyper Terminal)

When Hyper Terminal program is running, a window on monitor screen will display echo sounder information in various format of terminal information.
6) Echo sounder data collection
- Move mouse cursor into the echogram window and use right bottom mouse click for data collection drop down menu.
- Select “Collecting Start” button, left bottom mouse click. The echo sounder data will automatically create new data recorded file using current date and time as file name on Desktop folder. The format of file name will start with “FishFinder” followed by Date such as “FishFinder_yyyymmdd_hhmmss.csv”.
- To stop data recording, select “Collecting Stop” button, and use left bottom mouse click.

7) Ship position data collection
- In Hyper Terminal window, select drop down menu of “Transfer”, Capture Text” and enter Capture Text folder and file name.
- Select “Start” button. The echo sounder information text will automatically be recorded into defined data file name.
To stop data recording, select “Transfer”, Capture Text” and “Stop”

ECHO SOUNDER DATA ANALYSIS PROCEDURE

The Echo Sounder Data Collection System (AQFI-1301) was designed to display echogram (Figure 17) and data recorded digital echo signal (50 kHz) into PC screen and hard disc. The Echo Sounder Data Collection System was set up at fixed sampling rate of 0.0512 msec with total 800 sampling echo data of digital value echo-signal recorded into the PC hard disc for each transmitting ping.

Collecting data specification

Sampling rate = 0.0512 msec = 3.84 cm
Depth range = sampling rate x 800 samplings = 40.96 ms
Maximum depth = 40.96 x 0.75 = 30.72 m

Figure 17.  Echogram and A-scope display pattern shown on PC screen.
The echo sounder data can be analyzed using Excel program. This manual shows the procedure of analyzing data with Microsoft Excel.

1) Open echo sounder raw data file using the Excel program, then a table like Figure 18 is displayed. Column A indicates the recording date and time. The echo sounder data are set from column B to column ADU (800 data). Since the sampling rate of the system is 0.512 msec (and sampling frequency is 19.53 kHz), the echo data are recorded with 3.84 cm resolution as shown by the following equation:

\[
\text{Resolution} = \frac{\text{Sampling Rate} \times \text{Sound Speed}}{2} = \frac{0.512 \text{ (msec)} \times 1500 \text{ (m/sec)}}{2} = 3.84 \text{ (cm)}
\]

And the maximum depth that the system can measure is:

\[
\text{Max. depth} = \text{Resolution} \times \text{Amount of Data} = 3.84 \text{ cm} \times 800 = 30.72 \text{ m}
\]

The echo sounder FURUNO GP-1670F sends pulses of 50 and 200 kHz alternately. Then, recording echoes are also arranged alternately by 50 and 200 kHz data in the direction of the row. The return echo signal recorded by Echo Sounder Data Collection System (AQFI-1301) on the “.csv” Excel file will also be composed of rows of digital data information of 50 and 200 kHz alternately. However, Echo Sounder Data Collection

Figure 18. Echo sounder raw data.
System (AQFI-1301) is designed to analyze only 50-kHz data. The following processes show the way to filter the 50-kHz data.

2) Input “=AVERAGE(B1:ADU1)” in cell ADV1 for calculating the average value from cell B1 to ADU1 (Figure 19).

3) Copy cell ADV1. Select the whole column ADV and paste the formula (Figure 19). Then the low and high average values are being alternately shown in column ADV.

4) Select cell ADV1 and click “Data” > “Filter” of the upper menu (Figure 20).

5) Click the arrow in column ADV1, and then click “Number Filters”. Select “Greater Than Or Equal To…” (Figure 21), then the Custom Auto Filter box is opened. In the Custom AutoFilter box, type the criteria for filtering your data. For example, type 500 in the upper-right box to show the data higher than 500. Click “OK” to apply the filter (Figure 22).
Figure 20. Activation of the “Filter” tool.
Figure 21. Selecting the way of filtering data.

Figure 22. Custom AutoFilter box.
6) Copy the filtered data (column A to ADV) and paste to a new sheet (the sheet is written as “Sheet 2” below) (Figure 23).

The data should be converted to voltage values to evaluate the echo data. The equation for converting is:
VR = 0.00013 × Raw data value

7) Make an additional new sheet (the sheet is written as “Sheet 3” below). Copy column A of “Sheet 2” and paste to column A of “Sheet 3”. Select cell B1 of “Sheet 3” and input “=Sheet2!B1*0.00013” (Figure 24). Sheet 2 should be renamed, then input the name of the “renamed Sheet 2” instead of just Sheet 2 in the equation. Copy cell B1 of “Sheet 3”. Select cell B1 to cell ADU x (where x is the number of rows entered in the 50-kHz data in “Sheet 2”), and then paste.

Figure 23. Copying and pasting the filtered data to a new sheet.
To show the relationship between the voltage data and depth, make a graph as shown in the following processes.

8) Select row 1. Right-click and select “Insert” (Figure 25).

9) Input “Depth” in cell A1 and depth value (meter) in cell B1 to ADU1 (from left cell, 0, 0.0384, 0.0768, … ). Insert a “Scatter” graph.

10) Right-click on the graph and select “Select data …”, then the Select Data Source box is opened (Figure 26). Click “Add” in the Select Data Source box. If there is any data series, remove all. And then, Edit Series box is opened. Input Series name, Series x values, and
Series y values. For example, to show the graph of data in row 2, input the data as shown in Figure 27, and the graph like in Figure 28 is displayed.

Figure 25. Inserting a new row.

Figure 26. Selecting Data Source.
Figure 27. Example of an input data.

Figure 28. Graph showing the relationship between voltage values and depths.
In order to find out the fish abundance distribution in the survey area, a further echo signal data analyzing process is required as shown in the following:

1) Convert digital data (raw data) into voltage data (in terms of linear and decibels), (Figure 29),
2) Calculate the Time Varied Gain (TVG) compensation of loss echo signal for each layer of water depth (Figure 30),
3) Calculate the average value of echo signal for each transmitting ping, (Figure 31),
4) Calculate the integration value of echo signal of each integration distant, (Figure 32),
5) Plot the echo integration value in the survey transect for fish abundance distribution (Figure 33).

**Figure 29.** Converting digital data (raw data) in to voltage data.
Figure 30. Calculation of Time Varied Gain (TVG) compensation of loss echo signal for each layer of water depth.
Figure 31. Calculation of average value of echo signal for each transmitting ping.

Figure 32. Calculation of integration value of echo signal of each integration distance.
Figure 33. Plot of the echo integration value in the survey transect for fish abundance distribution.
GPS DATA ANALYSIS PROCEDURE

Outline of GPS data

The GPS receiver output data or record tracks, the time, accuracy, etc. in NMEA format. NMEA is acronym for the National Marine Electronics Association. There are different types of NMEA messages. Some of those that are applicable to GPS receivers are listed below.

- **GPGGA**: essential fix data which provides 3D location and accuracy data.
- **GPRMC**: has its own version of essential GPS pvt (position, velocity, time) data.
- **GPDPT**: Depth
- **GPVTG**: Speed over ground and tracking offset.

The GPGGA message has enough information for this Manual’s system. To understand the NMEA message structure, examine the $GPGGA message as shown below, this particular message was an output from GPS receiver:

$GPGGA,031959,1235.0071,N,10122.5079,E,1,16,0.7,3.6,M,,,,,*0C$

031959 is the time stamp: UTC time in hours, minutes and seconds.
1235.0071 is the latitude in the DDMM.MMMM format.
N denotes north latitude.
10122.5079 is the longitude in the DDDMM.MMMM format.
E denotes east longitude.
1 denotes the fixed quality: 1 = independent
2 = Differentially correct coordinate (e.g., WAAS, DGPS)
3 = PPS fix
4 = RTK fix coordinate (centimeter precision)
5 = RTK Float (decimeter precision).
16 denotes number of satellites used in the coordinate.
0.7 denotes the HDOP (horizontal dilution of precision).
3.6 denotes altitude, meters, above mean sea level.
M denotes units of altitude (eg. meters or feet)

(empty field): Height of geoid (mean sea level) above WGS84 ellipsoid

(empty field) time in seconds since last DGPS update

(empty field) DGPS station ID number

*0c is the checksum data, always begins with *

The NMEA of GPS is saved in text format. Here, the resource distribution is mapped with Microsoft Excel.

a. Loading NMEA data with Excel
   Open text file data by using Excel program
   File -> Open --> All Files
   At Text Import Wizard – Step 1 of 3, select “Fix width”, “Next”
b. At Text Import Wizard – Step 2 of 3, separate cells by hour, minute, second, degree, min., lat., degree, minute, and long., and select “Next”

c. At Text Import Wizard – Step 3 of 3, select “General”, “Finish”
d. The Excel data sheet will be displayed as follows:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SIGGA</td>
<td>2</td>
<td>S1</td>
<td>31</td>
<td>&quot;&quot;</td>
<td>12</td>
<td>34</td>
<td>8355</td>
<td>&quot;&quot;</td>
<td>101</td>
<td>22</td>
<td>7301</td>
<td>&quot;E,1</td>
</tr>
<tr>
<td>2</td>
<td>SIGMC</td>
<td>2</td>
<td>S1</td>
<td>31</td>
<td>&quot;&quot;</td>
<td>&quot;A&quot;</td>
<td>1234</td>
<td>8355</td>
<td>&quot;N&quot;</td>
<td>1122</td>
<td>7</td>
<td>&quot;301,E,2,3,278,5,210414&quot;,A*62&quot;</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SIIDPT</td>
<td>8</td>
<td>&quot;3&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;06&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>4</td>
<td>SIIZDA</td>
<td>2</td>
<td>S1</td>
<td>31</td>
<td>&quot;&quot;</td>
<td>21</td>
<td>04</td>
<td>20</td>
<td>14&quot;</td>
<td>4&quot;</td>
<td>07</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>5</td>
<td>SIIVTG</td>
<td>27</td>
<td>2</td>
<td>1</td>
<td>&quot;1&quot;</td>
<td>1,9</td>
<td>N</td>
<td>3,5</td>
<td>K</td>
<td>A5F</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>SIGGA</td>
<td>2</td>
<td>S1</td>
<td>32</td>
<td>&quot;&quot;</td>
<td>12</td>
<td>34</td>
<td>8354</td>
<td>&quot;&quot;</td>
<td>101</td>
<td>22</td>
<td>7289</td>
<td>&quot;E,1</td>
</tr>
<tr>
<td>7</td>
<td>SIGMC</td>
<td>2</td>
<td>S1</td>
<td>32</td>
<td>&quot;&quot;</td>
<td>&quot;A&quot;</td>
<td>1234</td>
<td>8354</td>
<td>&quot;N&quot;</td>
<td>1122</td>
<td>7</td>
<td>&quot;295,E,1,272,1,210414&quot;,A*68&quot;</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>SIIDPT</td>
<td>8</td>
<td>&quot;1&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;06&quot;</td>
<td>&quot;65&quot;</td>
<td>&quot;66&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>SIIZDA</td>
<td>2</td>
<td>S1</td>
<td>33</td>
<td>&quot;&quot;</td>
<td>21</td>
<td>04</td>
<td>20</td>
<td>14&quot;</td>
<td>4&quot;</td>
<td>07</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>10</td>
<td>SIIVTG</td>
<td>25</td>
<td>6</td>
<td>3</td>
<td>&quot;T&quot;</td>
<td>2,3</td>
<td>N</td>
<td>4,7</td>
<td>K</td>
<td>A53</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>SIGGA</td>
<td>2</td>
<td>S1</td>
<td>33</td>
<td>&quot;&quot;</td>
<td>12</td>
<td>34</td>
<td>8351</td>
<td>&quot;&quot;</td>
<td>101</td>
<td>22</td>
<td>7289</td>
<td>&quot;E,1</td>
</tr>
<tr>
<td>12</td>
<td>SIGMC</td>
<td>2</td>
<td>S1</td>
<td>33</td>
<td>&quot;&quot;</td>
<td>&quot;A&quot;</td>
<td>1234</td>
<td>8351</td>
<td>&quot;N&quot;</td>
<td>1122</td>
<td>7</td>
<td>&quot;295,E,1,272,1,210414&quot;,A*68&quot;</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>SIIDPT</td>
<td>8</td>
<td>&quot;0&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;06&quot;</td>
<td>&quot;64&quot;</td>
<td>&quot;66&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>SIIZDA</td>
<td>2</td>
<td>S1</td>
<td>33</td>
<td>&quot;&quot;</td>
<td>21</td>
<td>04</td>
<td>20</td>
<td>14&quot;</td>
<td>4&quot;</td>
<td>07</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>15</td>
<td>SIIVTG</td>
<td>25</td>
<td>9</td>
<td>0</td>
<td>&quot;T&quot;</td>
<td>1,8</td>
<td>N</td>
<td>3,3</td>
<td>K</td>
<td>A50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>SIGGA</td>
<td>2</td>
<td>S1</td>
<td>34</td>
<td>&quot;&quot;</td>
<td>12</td>
<td>34</td>
<td>8350</td>
<td>&quot;&quot;</td>
<td>101</td>
<td>22</td>
<td>7284</td>
<td>&quot;E,1</td>
</tr>
<tr>
<td>17</td>
<td>SIGMC</td>
<td>2</td>
<td>S1</td>
<td>34</td>
<td>&quot;&quot;</td>
<td>&quot;A&quot;</td>
<td>1234</td>
<td>8350</td>
<td>&quot;N&quot;</td>
<td>1122</td>
<td>7</td>
<td>&quot;284,E,1,1,259,0,210414&quot;,A*60&quot;</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>SIIDPT</td>
<td>8</td>
<td>&quot;3&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;06&quot;</td>
<td>&quot;67&quot;</td>
<td>&quot;68&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>SIIZDA</td>
<td>2</td>
<td>S1</td>
<td>35</td>
<td>&quot;&quot;</td>
<td>21</td>
<td>04</td>
<td>20</td>
<td>14&quot;</td>
<td>4&quot;</td>
<td>07</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>20</td>
<td>SIIVTG</td>
<td>24</td>
<td>6</td>
<td>3</td>
<td>&quot;T&quot;</td>
<td>1,9</td>
<td>N</td>
<td>3,6</td>
<td>K</td>
<td>A52</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>SIGGA</td>
<td>2</td>
<td>S1</td>
<td>35</td>
<td>&quot;&quot;</td>
<td>12</td>
<td>34</td>
<td>8347</td>
<td>&quot;&quot;</td>
<td>101</td>
<td>22</td>
<td>7279</td>
<td>&quot;E,1</td>
</tr>
<tr>
<td>22</td>
<td>SIGMC</td>
<td>2</td>
<td>S1</td>
<td>35</td>
<td>&quot;&quot;</td>
<td>&quot;A&quot;</td>
<td>1234</td>
<td>8347</td>
<td>&quot;N&quot;</td>
<td>1122</td>
<td>7</td>
<td>&quot;270,E,1,9,246,8,210414&quot;,A*62&quot;</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>SIIDPT</td>
<td>8</td>
<td>&quot;2&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;06&quot;</td>
<td>&quot;66&quot;</td>
<td>&quot;66&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>SIIZDA</td>
<td>2</td>
<td>S1</td>
<td>36</td>
<td>&quot;&quot;</td>
<td>21</td>
<td>04</td>
<td>20</td>
<td>14&quot;</td>
<td>4&quot;</td>
<td>07</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>25</td>
<td>SIIVTG</td>
<td>23</td>
<td>3</td>
<td>1</td>
<td>&quot;00&quot;</td>
<td>&quot;06&quot;</td>
<td>&quot;67&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;07&quot;</td>
<td>&quot;00&quot;</td>
<td>&quot;00&quot;</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>SIGGA</td>
<td>2</td>
<td>S1</td>
<td>36</td>
<td>&quot;&quot;</td>
<td>12</td>
<td>34</td>
<td>8343</td>
<td>&quot;&quot;</td>
<td>101</td>
<td>22</td>
<td>7276</td>
<td>&quot;E,1</td>
</tr>
<tr>
<td>27</td>
<td>SIGMC</td>
<td>2</td>
<td>S1</td>
<td>36</td>
<td>&quot;&quot;</td>
<td>&quot;A&quot;</td>
<td>1234</td>
<td>8343</td>
<td>&quot;N&quot;</td>
<td>1122</td>
<td>7</td>
<td>&quot;276,E,1,8,232,7,210414&quot;,A*67&quot;</td>
<td></td>
</tr>
</tbody>
</table>

e. Move cursor into cell which indicates "$IIGGA" where information on time, and position (Latitude and Longitude) are contained, use right bottom mouse click for drop down menu. Select “Filter”, “Filter by Selected Cell’s Value”. 
f. Excel data sheet will display only the row of data starting with “SIIGGA”. Column defines C as hour, D as minute, E as second, G as degree of latitude, H as minute of latitude, J as degree of longitude, K as minute of longitude.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
<td>SIIGGA</td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>31</td>
<td>12</td>
<td>34.8355</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7301</td>
<td>&quot;E&quot;,1,09,1,1,10.7,M,...06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>32</td>
<td>12</td>
<td>34.8354</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7295</td>
<td>&quot;E&quot;,1,09,1,1,0.3,M,...0C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>33</td>
<td>12</td>
<td>34.8351</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7289</td>
<td>&quot;E&quot;,1,09,1,1,0.7,M,...01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>34</td>
<td>12</td>
<td>34.835</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7284</td>
<td>&quot;E&quot;,1,09,1,1,0.8,M,...05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>35</td>
<td>12</td>
<td>34.8347</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7279</td>
<td>&quot;E&quot;,1,09,1,1,0.6,M,...0E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>36</td>
<td>12</td>
<td>34.8343</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7276</td>
<td>&quot;E&quot;,1,09,1,1,1.1,M,...00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>37</td>
<td>12</td>
<td>34.834</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7273</td>
<td>&quot;E&quot;,1,09,1,1,1.4,M,...02</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>38</td>
<td>12</td>
<td>34.8335</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7271</td>
<td>&quot;E&quot;,1,09,1,1,1.4,M,...00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>39</td>
<td>12</td>
<td>34.833</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7269</td>
<td>&quot;E&quot;,1,09,1,1,1.6,M,...02</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>40</td>
<td>12</td>
<td>34.8325</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7268</td>
<td>&quot;E&quot;,1,09,1,1,1.6,M,...00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>41</td>
<td>12</td>
<td>34.832</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7268</td>
<td>&quot;E&quot;,1,09,1,1,1.6,M,...00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>42</td>
<td>12</td>
<td>34.8315</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7268</td>
<td>&quot;E&quot;,1,09,1,1,1.4,M,...0A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>43</td>
<td>12</td>
<td>34.8309</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7268</td>
<td>&quot;E&quot;,1,09,1,1,1.7,M,...05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>44</td>
<td>12</td>
<td>34.8304</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7272</td>
<td>&quot;E&quot;,1,09,1,1,1.3,M,...02</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>45</td>
<td>12</td>
<td>34.8294</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7275</td>
<td>&quot;E&quot;,1,09,1,1,1.5,M,...0E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>46</td>
<td>12</td>
<td>34.8289</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7279</td>
<td>&quot;E&quot;,1,09,1,1,1.6,M,...09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>47</td>
<td>12</td>
<td>34.8289</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7279</td>
<td>&quot;E&quot;,1,09,1,1,1.6,M,...09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>48</td>
<td>12</td>
<td>34.8289</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7281</td>
<td>&quot;E&quot;,1,09,1,1,1.4,M,...0F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>49</td>
<td>12</td>
<td>34.8281</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7287</td>
<td>&quot;E&quot;,1,09,1,1,1.7,M,...0F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>50</td>
<td>12</td>
<td>34.8278</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7292</td>
<td>&quot;E&quot;,1,09,1,1,1.3,M,...01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>51</td>
<td>12</td>
<td>34.8274</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7298</td>
<td>&quot;E&quot;,1,09,1,1,1.5,M,...00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>52</td>
<td>12</td>
<td>34.8274</td>
<td>&quot;N&quot;</td>
<td>101</td>
<td>22.7298</td>
<td>&quot;E&quot;,1,09,1,1,1.5,M,...00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Copy all the data to a new work sheet.

g. Delete data in column A, B, F, I and L-N.
h. After deleting, calculate UTC (Coordinated Universal Time) and degree of latitude and longitude, column H as 1:00, I as 0:01, J as 00:00:01, K as $=+H1*A1+B1*I1+C1*J1$, L as $=+D1+E1/60$ and M as $=+F1+G1/60$.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>S1</td>
<td>31</td>
<td>12</td>
<td>34.8355</td>
<td>101</td>
<td>22.7301</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58059167</td>
<td>101.378835</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>S1</td>
<td>32</td>
<td>12</td>
<td>34.8354</td>
<td>101</td>
<td>22.7295</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58058233</td>
<td>101.378825</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>S1</td>
<td>33</td>
<td>12</td>
<td>34.8351</td>
<td>101</td>
<td>22.7289</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58057331</td>
<td>101.378815</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>S1</td>
<td>34</td>
<td>12</td>
<td>34.8345</td>
<td>101</td>
<td>22.7284</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58056467</td>
<td>101.378807</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>S1</td>
<td>35</td>
<td>12</td>
<td>34.8347</td>
<td>101</td>
<td>22.7279</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58055833</td>
<td>101.378798</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>S1</td>
<td>36</td>
<td>12</td>
<td>34.8343</td>
<td>101</td>
<td>22.7276</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58055167</td>
<td>101.378793</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>S1</td>
<td>37</td>
<td>12</td>
<td>34.8339</td>
<td>101</td>
<td>22.7273</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58054667</td>
<td>101.378783</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>S1</td>
<td>38</td>
<td>12</td>
<td>34.8335</td>
<td>101</td>
<td>22.7271</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58054255</td>
<td>101.378778</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>S1</td>
<td>39</td>
<td>12</td>
<td>34.8330</td>
<td>101</td>
<td>22.7269</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58053851</td>
<td>101.378771</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>S1</td>
<td>40</td>
<td>12</td>
<td>34.8325</td>
<td>101</td>
<td>22.7268</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58053467</td>
<td>101.378768</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>S1</td>
<td>41</td>
<td>12</td>
<td>34.8323</td>
<td>101</td>
<td>22.7266</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58053067</td>
<td>101.378767</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>S1</td>
<td>42</td>
<td>12</td>
<td>34.8318</td>
<td>101</td>
<td>22.7264</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58052667</td>
<td>101.378766</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>S1</td>
<td>43</td>
<td>12</td>
<td>34.8314</td>
<td>101</td>
<td>22.7263</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58052331</td>
<td>101.378763</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>S1</td>
<td>44</td>
<td>12</td>
<td>34.8308</td>
<td>101</td>
<td>22.7260</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58051967</td>
<td>101.378762</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>S1</td>
<td>45</td>
<td>12</td>
<td>34.8302</td>
<td>101</td>
<td>22.7259</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58051667</td>
<td>101.378761</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>S1</td>
<td>46</td>
<td>12</td>
<td>34.8297</td>
<td>101</td>
<td>22.7257</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58051367</td>
<td>101.378760</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>S1</td>
<td>47</td>
<td>12</td>
<td>34.8290</td>
<td>101</td>
<td>22.7255</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58051067</td>
<td>101.378759</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>S1</td>
<td>48</td>
<td>12</td>
<td>34.8283</td>
<td>101</td>
<td>22.7253</td>
<td>1:00</td>
<td>0:00:01</td>
<td>2:51</td>
<td>12.58049467</td>
<td>101.378758</td>
</tr>
</tbody>
</table>

Excel sheet UTC time will show column K

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>31</td>
<td>34.8355</td>
<td>101</td>
<td>22.7301</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:31</td>
<td>12.58059167</td>
<td>101.378035</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>32</td>
<td>34.8354</td>
<td>101</td>
<td>22.7295</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:32</td>
<td>12.580595</td>
<td>101.378825</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>33</td>
<td>34.8351</td>
<td>101</td>
<td>22.7289</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:33</td>
<td>12.580585</td>
<td>101.378815</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>34</td>
<td>34.835</td>
<td>101</td>
<td>22.7284</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:34</td>
<td>12.58058233</td>
<td>101.3788076</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>35</td>
<td>34.8347</td>
<td>101</td>
<td>22.7279</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:35</td>
<td>12.58057833</td>
<td>101.3787983</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>36</td>
<td>34.8344</td>
<td>101</td>
<td>22.7276</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:36</td>
<td>12.58057167</td>
<td>101.3787933</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>37</td>
<td>34.834</td>
<td>101</td>
<td>22.7277</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:37</td>
<td>12.58056667</td>
<td>101.3787883</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>38</td>
<td>34.8335</td>
<td>101</td>
<td>22.7271</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:38</td>
<td>12.58056633</td>
<td>101.3787895</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>39</td>
<td>34.833</td>
<td>101</td>
<td>22.7269</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:39</td>
<td>12.580565</td>
<td>101.3787817</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>40</td>
<td>34.8325</td>
<td>101</td>
<td>22.7268</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:40</td>
<td>12.58055167</td>
<td>101.3787867</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>41</td>
<td>34.832</td>
<td>101</td>
<td>22.7268</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:41</td>
<td>12.580551</td>
<td>101.3787867</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>42</td>
<td>34.8315</td>
<td>101</td>
<td>22.7268</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:42</td>
<td>12.5805525</td>
<td>101.3787867</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>43</td>
<td>34.8309</td>
<td>101</td>
<td>22.7268</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:43</td>
<td>12.5805515</td>
<td>101.3787867</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>44</td>
<td>34.8304</td>
<td>101</td>
<td>22.727</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:44</td>
<td>12.58055667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>45</td>
<td>34.8304</td>
<td>101</td>
<td>22.727</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:44</td>
<td>12.58055667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>46</td>
<td>34.8294</td>
<td>101</td>
<td>22.7275</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:46</td>
<td>12.58049075</td>
<td>101.3787917</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>47</td>
<td>34.8289</td>
<td>101</td>
<td>22.7279</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:47</td>
<td>12.58048167</td>
<td>101.3787983</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>48</td>
<td>34.828</td>
<td>101</td>
<td>22.7279</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:48</td>
<td>12.580476</td>
<td>101.3787983</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>49</td>
<td>34.8281</td>
<td>101</td>
<td>22.7278</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:49</td>
<td>12.58046833</td>
<td>101.3787817</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>50</td>
<td>34.8278</td>
<td>101</td>
<td>22.7292</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:50</td>
<td>12.58046333</td>
<td>101.3787882</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>51</td>
<td>34.8274</td>
<td>101</td>
<td>22.7298</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:51</td>
<td>12.58045667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>52</td>
<td>34.8274</td>
<td>101</td>
<td>22.7298</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:51</td>
<td>12.58045667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>53</td>
<td>34.8274</td>
<td>101</td>
<td>22.7298</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:51</td>
<td>12.58045667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>2</td>
<td>54</td>
<td>34.8274</td>
<td>101</td>
<td>22.7298</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:51</td>
<td>12.58045667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>55</td>
<td>34.8274</td>
<td>101</td>
<td>22.7298</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:51</td>
<td>12.58045667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>2</td>
<td>56</td>
<td>34.8274</td>
<td>101</td>
<td>22.7298</td>
<td>100</td>
<td>00:01</td>
<td>00:01</td>
<td>2.51:51</td>
<td>12.58045667</td>
<td>101.3787833</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


k. The column A is UTC. Normally, local mean time is used in the research. So the local time is shown in column E (UTC add Time difference). The columns of latitude and longitude should be reversed, because the latitude is on the y axis and longitude on the x axis.
1. Select Columns F and G, select insert chart, select a scatter plot chart. This procedure can produce a drawing of the track line.
m. Add shore line data and detect with the tracking data. This procedure will produces a rough survey map. After a chart is added, some of the default elements should be modified to create an exquisite eye-catching map.

Plotting of survey data cruise tract could be displayed to fit with the design survey transect in the study area. This experiment was conducted in the set-net fishing ground of Banphe, Rayong Province, Thailand. The coverage survey area is 2.5 x 6.5 square kilometers with parallel cruise tract of 500 meters apart.

Survey transect coverage in the set-net fishing ground of Banphe, Rayong Province, Thailand.
Source Code of Program “FishFinder, Version 3.2” in Visual C

<Original source code can be downloaded from RIHN archive database: http://www.chikyu.ac.jp/rihn/archive_database/archive/index.html>

/*****************************************************************************
* Program
*****************************************************************************/

using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace jPicoFishfinder1.0
{
    struct ChannelSettings
    {
        public bool DCcoupled;
        public Imports.Range range;
        public bool enabled;
    }

    class Pwq
    {
        public Imports.PwqConditions[] conditions;
        public short nConditions;
        public Imports.ThresholdDirection direction;
        public uint lower;
        public uint upper;
        public Imports.PulseWidthType type;

        public Pwq(Imports.PwqConditions[] conditions,
                   short nConditions,
                   Imports.ThresholdDirection direction,
                   uint lower, uint upper,
                   Imports.PulseWidthType type)
        {
            this.conditions = conditions;
            this.nConditions = nConditions;
            this.direction = direction;
            this.lower = lower;
            this.upper = upper;
            this.type = type;
        }
    }

    static class Program
    {
    }
}
/// <summary>
/// The main entry point for the application.
/// </summary>
[STAThread]
static void Main()
{
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);

    short handle;
    short status = Imports.OpenUnit(out handle);   //1. Open the oscilloscope
    if (status != 0)
    {
        MessageBox.Show("Unable to open device", "ERROR");
    }
    else
    {
        Application.Run(new Form1(handle));
        //7. Run the form
        Imports.CloseUnit(handle);   //12. Close the oscilloscope
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading;
using System.IO;
using Emgu.CV;
using Emgu.CV.Structure;
using System.Net;

namespace jPicoFishfinder1._0
{
    public partial class Form1 : Form
    {
        //colorbar for GPS version
        double[] m_dLevel = { 1000, 900, 850, 800, 750, 700, 650 };

        //fishfinder

        Thread m_threadcvImg;
        int m_cvImgStart = 0;
        Point m_pLeftTop;
        Point m_pRightBottom;
        Bitmap m_cvBmp;
        bool m_bOdd=true;

        //1.1.1
        short[] m_sPingData;

        //AD 2 SV
        double m_dSpeed ;// Double.Parse(txtSpeed.Text);
        double m_dSR ;// Double.Parse(txtSampRate.Text);
        double m_dAbsorb;// Double.Parse(txtAbsorb.Text);
        double m_dTau;// Double.Parse(txtPulseW.Text);
        double m_dVar; // m_dVar = 10 * Math.Log10(0.5 * m_dSpeed * m_dTau * m_dPhi) -
            m_dK;
    }
}
// pico below
private readonly short _handle;
public const int BUFFER_SIZE = 800;  // sample rage == 20k, set the range here, 1 sample == 0.0375m, 1m == 26.7 samples (for 1500m/s); so (30m: 800 samples, 40m: 1068 samples)

// 20150323
public const int MAX_CHANNELS = 4;
public const int QUAD_SCOPE = 4;
public const int DUAL_SCOPE = 2;

uint _timebase = 1001;
int _timeInterval;
bool _logFlag = false;
bool _saveFlag = false;

short _oversample = 1;
bool _scaleVoltages = true;

ushort[] inputRanges = { 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000, 20000, 50000 };
bool _ready = false;
short _trig = 0;
uint _trigAt = 0;
int _sampleCount = 0;
uint _startIndex = 0;
bool _autoStop;
private ChannelSettings[] _channelSettings;
private int _channelCount;

private Imports.Range _firstRange;
private Imports.Range _lastRange;
private Imports.ps4000BlockReady _callbackDelegate;
long second = 10;

Thread m_threadLoggin;

public Form1(short handle)
{
    _handle = handle;
    InitializeComponent();
    GetDeviceInfo();
    pictureBox_echograph.BackColor = Color.DarkBlue;
    // m_cvBmp = new Bitmap(pictureBox_echograph.Width, pictureBox_echograph.Height);
}

/**************************************************************************
* Initialise unit' structure with Variant specific defaults
**************************************************************************/
void GetDeviceInfo()
{
    int variant = 0;
}
string[] description = {
    "Driver Version   ",
    "USB Version      ",
    "Hardware Version ",
    "Variant Info     ",
    "Serial          ",
    "Cal Date        ",
    "Kernel Ver      
};

System.Text.StringBuilder line = new System.Text.StringBuilder(80);

if (_handle >= 0)
{
    string srLable2="";
    textBox1.Clear();
    for (int i = 0; i < 7; i++)
    {
        short requiredSize;
        Imports.GetUnitInfo(_handle, line, 80, out requiredSize, i);
        if (i == 3)
        {
            variant = Convert.ToInt16(line.ToString());
        }
        srLable2 = String.Format("{0}: {1}", description[i], line);
        srLable2 = srLable2 + "\n";
        textBox1.AppendText(srLable2);
    }
    srLable2 = String.Format("Sampling Rate     : {0} ksps", 1000000/(50*(_timebase-1)));
    textBox1.AppendText(srLable2);

    switch (variant)
    {
    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = DUAL_SCOPE;
        break;

    case (int)Imports.Model.PS4224:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_20V;
        _channelCount = DUAL_SCOPE;
        break;

    case (int)Imports.Model.PS4423:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;

    case (int)Imports.Model.PS4223:
        _firstRange = Imports.Range.Range_50MV;
        _lastRange = Imports.Range.Range_100V;
        _channelCount = QUAD_SCOPE;
        break;

    case (int)Imports.Model.PS4424:
        _firstRange = Imports.Range.Range_50MV;
_lastRange = Imports.Range.Range_20V;
_channelCount = QUAD_SCOPE;
break;

case (int)Imports.Model.PS4262:
    _firstRange = Imports.Range.Range_10MV;
    _lastRange = Imports.Range.Range_20V;
    _channelCount = DUAL_SCOPE;
    break;
/
/***************************************************************************/
/* Callback */
/* used by PS4000 data block collection calls, on receipt of data. */
/* used to set global flags etc checked by user routines */
/***************************************************************************/
void BlockCallback(short handle, short status, IntPtr pVoid)
{
    // flag to say done reading data
    _ready = true;
}

/***************************************************************************/
/* SetDefaults - restore default settings */
/***************************************************************************/
void SetDefaults()
{
    for (int i = 0; i < _channelCount; i++) // reset channels to most recent settings
    {
        Imports.SetChannel(_handle, Imports.Channel.ChannelA + i,
            (short)(_channelSettings[(int)(Imports.Channel.ChannelA + i)].enabled ? 1 : 0),
            (short)(_channelSettings[(int)(Imports.Channel.ChannelA + i)].DCcoupled ? 1 : 0),
        }
```c
*_channelSettings[(int)(Imports.Channel.ChannelA + i)].range;
}
}

/******************************************************************************
* Select _timebase, set _oversample to on and time units as nano seconds
******************************************************************************/
void SetTimebase(uint timebase)
{
    _timebase = timebase;
    int maxSamples;

    while (Imports.GetTimebase(_handle, _timebase, BUFFER_SIZE, out _timeInterval, 1, out maxSamples, 0) != 0)
    {
        _timebase++;
    }
    _oversample = 1;
}

/******************************************************************************
* SetTrigger
* this function sets all the required trigger parameters, and calls the
* triggering functions
******************************************************************************/
{
    short status;

    if ((status = Imports.SetTriggerChannelProperties(_handle, channelProperties, nChannelProperties, auxOutputEnabled, autoTriggerMs)) != 0)
    {
        return status;
    }

    if ((status = Imports.SetTriggerChannelConditions(_handle, triggerConditions, nTriggerConditions)) != 0)
    {
        return status;
    }
```
if (directions == null) directions = new Imports.ThresholdDirection[] {
    Imports.ThresholdDirection.None,
    Imports.ThresholdDirection.None, Imports.ThresholdDirection.None,
    Imports.ThresholdDirection.None,

if ((status = Imports.SetTriggerChannelDirections(_handle,
    directions[(int)Imports.Channel.ChannelA],
    directions[(int)Imports.Channel.ChannelB],
    directions[(int)Imports.Channel.ChannelC],
    directions[(int)Imports.Channel.ChannelD],
    directions[(int)Imports.Channel.External],
    directions[(int)Imports.Channel.Aux])) != 0)
{
    return status;
}

if ((status = Imports.SetTriggerDelay(_handle, delay)) != 0)
{
    return status;
}

if (pwq == null) pwq = new Pwq(null, 0, Imports.ThresholdDirection.None, 0, 0,
    Imports.PulseWidthType.None);

status = Imports.SetPulseWidthQualifier(_handle, pwq.conditions,
    pwq.nConditions, pwq.direction,
    pwq.lower, pwq.upper, pwq.type);

return status;

/***************************************************************************/
/* adc_to_mv */
/* Convert an 16-bit ADC count into millivolts */
/***************************************************************************/
int adc_to_mv(int raw, int ch)
{
    return (raw * inputRanges[ch]) / Imports.MaxValue;
}

/***************************************************************************/
/* mv_to_adc */
/* Convert a millivolt value into a 16-bit ADC count */
/* (useful for setting trigger thresholds) */

59
short mv_to_adc(short mv, short ch)
{
    return (short)((mv * Imports.MaxValue) / inputRanges[ch]);
}

/* CollectBlockTrigged *
* this function demonstrates how to collect a single block of data from the *
* unit, when a trigger event occurs. *
* */
void SetBlockTrigger()
{
    short triggerVoltage = mv_to_adc(1000,
    (short)_channelSettings[(int)Imports.Channel.ChannelA].range); // ChannelInfo stores ADC counts
    Imports.TriggerChannelProperties[] sourceDetails = new Imports.TriggerChannelProperties[] {
        new Imports.TriggerChannelProperties(triggerVoltage,
            256*10,
            triggerVoltage,
            256*10,
            Imports.Channel.ChannelA,
            Imports.ThresholdMode.Level)};

    Imports.TriggerConditions[] conditions = new Imports.TriggerConditions[] {
        new Imports.TriggerConditions(Imports.TriggerState.True,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare,
            Imports.TriggerState.DontCare)};

    Imports.ThresholdDirection[] directions = new Imports.ThresholdDirection[] {
        Imports.ThresholdDirection.Rising,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None,
        Imports.ThresholdDirection.None};
    SetDefaults();
    /* Trigger enabled *
    * Rising edge *
    * Threshold = 1000mV */
    SetTrigger(sourceDetails, 1, conditions, 1, directions, null, 0, 0, 0); //4. SetTrigger
}
/****************************************************************************
* DataLoggin - Logging Data to file
****************************************************************************/
public void DataLoggin()
{
    // setup devices
    //            _timebase = 1;

    _channelSettings = new ChannelSettings[MAX_CHANNELS];

    uint sampleCount = BUFFER_SIZE;
    PinnedArray<short>[] bufPinned = new PinnedArray<short>[1];

    int timeIndisposed;
    for (int i = 0; i < MAX_CHANNELS; i++)
    {
        _channelSettings[i].enabled = true;
        _channelSettings[i].DCcoupled = true;
        _channelSettings[i].range = Imports.Range.Range_5V;
    }

    SetDefaults(); //2. Select channel ranges and AC/DC coupling
    SetTimebase(1001); //3. Select timebase until the required nanoseconds per sample is located
    SetBlockTrigger(); //4. Use the trigger setup functions to set up the trigger

    short[] buffers = new short[sampleCount];
    bufPinned[0] = new PinnedArray<short>(buffers);
    int status = Imports.SetDataBuffer(_handle, (Imports.Channel)1, buffers, (int)sampleCount);//7. tell the driver where your memory buffer is

    if (!Directory.Exists("C:\FishloggerData"))
    {
        Directory.CreateDirectory("C:\FishloggerData");
    }
    TextWriter writer = new StreamWriter("C:\FishloggerData\" + DateTime.Now.ToString("yyyyMMdd_HHmmss") + ".csv", false); //hh 12hour; HH 24hour

    //1.1 PingData
    m_sPingData = new short[sampleCount];

    double dLogData;
    double dShowData;

    while (_logFlag == true)
    {
        _ready = false;
    }
_callbackDelegate = BlockCallback;
//5. Start the oscilloscope running
Imports.RunBlock(_handle, 0, (int)sampleCount, _timebase, _oversample, out timelndisposed, 0, _callbackDelegate, IntPtr.Zero);

while (!_ready) //6. wait until the oscilloscope is ready using the ps4000BlockReady Callback
{
   Thread.Sleep(100);
}

//*************write data to file******************
if (_ready) //
{
   short overflow;
   Imports.GetValues(_handle, 0, ref sampleCount, 1, Imports.DownSamplingMode.None, 0, out overflow); //8. Transfer the block of data from the ocllscope
   writer.Write(DateTime.Now.ToString("yyyyMMdd_HH:mm:ss,"));
   for (int i = 0; i < sampleCount; i++) //9 handle the data
   {

      dLogData = adc_to_mv(bufPinned[0].Target[i], (short)_channelSettings[(int)Imports.Channel.ChannelB].range); //voltage
      // dLogData = bufPinned[0].Target[i];      //AD
      dShowData = dLogData;
      writer.Write("{0}", dLogData);

      if (dShowData > m_dLevel[0])
         m_sPingData[i] = 7;
      else if (dShowData > m_dLevel[1])
         m_sPingData[i] = 6;
      else if (dShowData > m_dLevel[2])
         m_sPingData[i] = 5;
      else if (dShowData > m_dLevel[3])
         m_sPingData[i] = 4;
      else if (dShowData > m_dLevel[4])
         m_sPingData[i] = 3;
      else if (dShowData > m_dLevel[5])
         m_sPingData[i] = 2;
      else if (dShowData > m_dLevel[6])
         m_sPingData[i] = 1;
      else
          m_sPingData[i] = 0;

   }
   writer.WriteLine();

   //draw one Ping
//************************
//draw by Main thread using Invoker
MethodInvoker mi = new MethodInvoker(this.drawEchograph);
this.BeginInvoke(mi);

/**************************
} //10. repeat steps 5-9

_logFlag = false;
Imports.Stop(_handle); //11.Stop the oscilloscope
writer.Close();
foreach (PinnedArray<short> p in bufPinned)
{
    if (p != null)
        p.Dispose();
}
_saveFlag = true;

private void btnDataLoggin_Click(object sender, EventArgs e)
{
    timer1.Enabled = false;
    second = 0;
    label1.Text = new DateTime(second * 10000000).ToLocalTimeString();

    _logFlag = true;
    btnDataLoggin.Enabled = false;
    btnStopLoggin.Enabled = true;
    m_threadLoggin = new Thread(new ThreadStart(DataLoggin));
    m_threadLoggin.Start();
    timer2.Enabled = true;
    label3.Text = "Data is logging... ... ...";
}

private void btnStopLoggin_Click(object sender, EventArgs e)
{
    timer2.Enabled = false; //no auto logging anymore
    m_threadLoggin.Suspend();
    _logFlag = false;
    btnStopLoggin.Enabled = false;
    btnDataLoggin.Enabled = true;
    m_threadLoggin.Resume();
    label3.Text = "Data logging Over";
}
private void timer1_Tick(object sender, EventArgs e) // one second interval
{
    second--; // Subtract one second
    label1.Text = new DateTime(second*10000000).ToLongTimeString();

    if (second == 0)
    {
        timer1.Enabled = false;
        _logFlag = true;
        btnDataLoggin.Enabled = false;
        btnStopLoggin.Enabled = true;
        m_threadLoggin = new Thread(new ThreadStart(DataLoggin));
        m_threadLoggin.Start();
        timer2.Enabled = true;
        label3.Text = "Data is logging... ... ...";
    }
}

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    if (MessageBox.Show("Sure to Stop Logging before closing the Application", "Attention", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)
    {
    }
    else e.Cancel = true;
}

private void timer2_Tick(object sender, EventArgs e) // one hour interval auto logging file
{
    // ***********Close current file first***********
    m_threadLoggin.Suspend();
    _logFlag = false;
    btnStopLoggin.Enabled = false;
    btnDataLoggin.Enabled = true;
    m_threadLoggin.Resume();
    label3.Text = "Data logging Over";

    // ***********New file, restart thread***********
    while (_saveFlag == false); // wait for the terminating of current m_threadLogging
    _saveFlag = false; // clear
    _logFlag = true; // clear
    btnDataLoggin.Enabled = false;
    btnStopLoggin.Enabled = true;
    m_threadLoggin = new Thread(new ThreadStart(DataLoggin));
    m_threadLoggin.Start();
    label3.Text = "Data is logging... ... ...";
}

// JPic'oFishfinder 1.1 add functions as below
private void drawEchograph()
{
    uint isamples = BUFFER_SIZE;

    Bitmap bitmap = new Bitmap(this.pictureBox_echograph.Width,
                              this.pictureBox_echograph.Height);
    Graphics gEcho = Graphics.FromImage(bitmap);  // this is like doublebuffer, draw on the
    // bitmap, and show on the pictureBox

    float y = pictureBox_echograph.Height / (float)isamples;
    float x = this.pictureBox_echograph.Width - 1;  // gEcho.PageScale;
    //        int x =1311;  //((840*3.125/2)

    if (this.pictureBox_echograph.Image == null)
    {
        for (int i = 0; i < isamples; i++)
        {
            gEcho.DrawLine(new Pen(colorbar[m_sPingData[i]]), x, i * y, x, (i + 1) * y);
        }
    } else
    {
        gEcho.DrawImage(this.pictureBox_echograph.Image, -1, 0);  // draw image from
        // current picturebox to a new position Drawlmage(image,x,y),==translate
        //         gEcho.DrawImage(this.pictureBox_echograph.Image, new Rectangle(0, 0,
        pictureBox_echograph.Width-1, pictureBox_echograph.Height), new Rectangle(1, 0,
        this.pictureBox_echograph.Image.Width-1, this.pictureBox_echograph.Image.Height),
        GraphicsUnit.Pixel);
        for (int i = 0; i < isamples; i++)
        {
            gEcho.DrawLine(new Pen(colorbar[m_sPingData[i]]), x, i * y, x, (i + 1) * y);
        }
    //      DrawDepthLable(gEcho, irange);  // draw depth label
        this.pictureBox_echograph.Image = bitmap;  // update to the pictureBox

        gEcho.Dispose();
    }
}
namespace jPicoFishfinder1._0
{
    partial class Form1
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            System.ComponentModel.ComponentResourceManager resources = new System.ComponentModel.ComponentResourceManager(typeof(Form1));
            this.btnDataLoggin = new System.Windows.Forms.Button();
            this.btnStopLoggin = new System.Windows.Forms.Button();
            this.label1 = new System.Windows.Forms.Label();
            this.timer1 = new System.Windows.Forms.Timer(this.components);
            this.label2 = new System.Windows.Forms.Label();
            this.textBox1 = new System.Windows.Forms.TextBox();
            this.timer2 = new System.Windows.Forms.Timer(this.components);
            this.label3 = new System.Windows.Forms.Label();
            this.pictureBox_echograph = new System.Windows.Forms.PictureBox();
            this.pictureBox_Colorbar = new System.Windows.Forms.PictureBox();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox_echograph)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox_Colorbar)).BeginInit();
            this.SuspendLayout();
            //
            // btnDataLoggin
            //
            //
            //
        }
    }
}
this.btnDataLoggin.Location = new System.Drawing.Point(940, 247);
this.btnDataLoggin.Name = "btnDataLoggin";
this.btnDataLoggin.Size = new System.Drawing.Size(101, 37);
this.btnDataLoggin.TabIndex = 0;
this.btnDataLoggin.Text = "Start Logging";
this.btnDataLoggin.UseVisualStyleBackColor = true;
this.btnDataLoggin.Click += new System.EventHandler(this.btnDataLoggin_Click);

// btnStopLoggin
//
this.btnStopLoggin.Enabled = false;
this.btnStopLoggin.Location = new System.Drawing.Point(1055, 247);
this.btnStopLoggin.Name = "btnStopLoggin";
this.btnStopLoggin.Size = new System.Drawing.Size(98, 37);
this.btnStopLoggin.TabIndex = 1;
this.btnStopLoggin.Text = "Stop Logging";
this.btnStopLoggin.UseVisualStyleBackColor = true;
this.btnStopLoggin.Click += new System.EventHandler(this.btnStopLoggin_Click);

// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(954, 189);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(185, 40);
this.label1.TabIndex = 2;
this.label1.Text = "00:00:00";

// timer1
//
this.timer1.Enabled = true;
this.timer1.Interval = 1000;
this.timer1.Tick += new System.EventHandler(this.timer1_Tick);

// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(934, 9);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(103, 12);
this.label2.TabIndex = 3;
this.label2.Text = "Device Information:\r\n";

// textBox1
//
this.textBox1.Location = new System.Drawing.Point(936, 33);
this.textBox1.Multiline = true;
this.textBox1.Name = "textBox1";
this.textBox1.ReadOnly = true;
this.textBox1.Size = new System.Drawing.Size(226, 106);
this.textBox1.TabIndex = 4;
this.textBox1.Text = "Driver Version    
\nUSB Version       
\nHardware Version  
\nVariant Info   
\nSerial            
\nCal Date          
\nKernel Ver        
\nSampling" + 
" Rate     : 

// timer2
//
this.timer2.Interval = 3600000;
this.timer2.Tick += new System.EventHandler(this.timer2_Tick);

// label3
//
this.label3.AutoSize = true;
this.label3.Font = new System.Drawing.Font("SimSun", 12F,
this.label3.Location = new System.Drawing.Point(937, 157);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(216, 16);
this.label3.TabIndex = 5;
this.label3.Text = "Data logging will start...";

// pictureBox_echograph
//
this.pictureBox_echograph.Location = new System.Drawing.Point(13, 13);
this.pictureBox_echograph.Name = "pictureBox_echograph";
this.pictureBox_echograph.Size = new System.Drawing.Size(904, 676);
this.pictureBox_echograph.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;

// pictureBox_Colorbar
//
using System;
using System.Runtime.InteropServices;
using System.Text;

namespace jPicoFishfinder1._0
{

class Imports
{
    #region constants

    private const string _DRIVER_FILENAME = "ps4000.dll";

    public const int MaxValue = 32764;
    #endregion

    #region Driver enums

    public enum Channel : int
    {
        ChannelA,
        ChannelB,
    }

    #endregion
}
public enum Range : int {
    Range_10MV, 
    Range_20MV, 
    Range_50MV, 
    Range_100MV, 
    Range_200MV, 
    Range_500MV, 
    Range_1V, 
    Range_2V, 
    Range_5V, 
    Range_10V, 
    Range_20V, 
    Range_50V, 
    Range_100V,
}

public enum ReportedTimeUnits : int {
    FemtoSeconds, 
    PicoSeconds, 
    NanoSeconds, 
    MicroSeconds, 
    MilliSeconds, 
    Seconds,
}

public enum ThresholdMode : int {
    Level, 
    Window
}

public enum ThresholdDirection : int {
    // Values for level threshold mode 
    // 
    Above,
Below,
Rising,
Falling,
RisingOrFalling,

// Values for window threshold mode
//
Inside = Above,
Outside = Below,
Enter = Rising,
Exit = Falling,
EnterOrExit = RisingOrFalling,

None = Rising,
}

public enum DownSamplingMode : int
{
    None,
    Aggregate
}

public enum PulseWidthType : int
{
    None,
    LessThan,
    GreaterThan,
    InRange,
    OutOfRange
}

public enum TriggerState : int
{
    DontCare,
    True,
    False,
}

public enum Model : int
{
    NONE = 0,
    PS4223 = 4223,
    PS4224 = 4224,
    PS4423 = 4423,
    PS4424 = 4424,
    PS4226 = 4226,
    PS4227 = 4227,
    PS4262 = 4262,
public struct TriggerChannelProperties
{
    public short ThresholdMajor;
    public ushort HysteresisMajor;
    public short ThresholdMinor;
    public ushort HysteresisMinor;
    public Channel Channel;
    public ThresholdMode ThresholdMode;

    public TriggerChannelProperties(
        short thresholdMajor,
        ushort hysteresisMajor,
        short thresholdMinor,
        ushort hysteresisMinor,
        Channel channel,
        ThresholdMode thresholdMode)
    {
        this.ThresholdMajor = thresholdMajor;
        this.HysteresisMajor = hysteresisMajor;
        this.ThresholdMinor = thresholdMinor;
        this.HysteresisMinor = hysteresisMinor;
        this.Channel = channel;
        this.ThresholdMode = thresholdMode;
    }
}

public struct TriggerConditions
{
    public TriggerState ChannelA;
    public TriggerState ChannelB;
    public TriggerState ChannelC;
    public TriggerState ChannelD;
    public TriggerState External;
    public TriggerState Aux;
    public TriggerState Pwq;

    public TriggerConditions(
        TriggerState channelA,
        TriggerState channelB,
        TriggerState channelC,
        TriggerState channelD,
        TriggerState external,
        TriggerState aux,
        TriggerState pwq)
    {
        this.ChannelA = channelA;
    }
this.ChannelB = channelB;
this.ChannelC = channelC;
this.ChannelD = channelD;
this.External = external;
this.Aux = aux;
this.Pwq = pwq;

```csharp
[StructLayout(LayoutKind.Sequential, Pack = 1)]
public struct PwqConditions
{
    public TriggerState ChannelA;
    public TriggerState ChannelB;
    public TriggerState ChannelC;
    public TriggerState ChannelD;
    public TriggerState External;
    public TriggerState Aux;

    public PwqConditions(
        TriggerState channelA,
        TriggerState channelB,
        TriggerState channelC,
        TriggerState channelD,
        TriggerState external,
        TriggerState aux)
    {
        this.ChannelA = channelA;
        this.ChannelB = channelB;
        this.ChannelC = channelC;
        this.ChannelD = channelD;
        this.External = external;
        this.Aux = aux;
    }
}
```
int noOfSamples,
short overflow,
uint triggerAt,
short triggered,
IntPtr pVoid);
#endregion

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000OpenUnit")]
public static extern short OpenUnit(out short handle);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000CloseUnit")]
public static extern short CloseUnit(short handle);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000RunBlock")]
public static extern short RunBlock(
    short handle,
    int noOfPreTriggerSamples,
    int noOfPostTriggerSamples,
    uint timebase,
    short oversample,
    out int timeIndisposedMs,
    ushort segmentIndex,
    ps4000BlockReady lpps4000BlockReady,
    IntPtr pVoid);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000Stop")]
public static extern short Stop(short handle);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetChannel")]
public static extern short SetChannel(
    short handle,
    Channel channel,
    short enabled,
    short dc,
    Range range);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetDataBuffers")]
public static extern short SetDataBuffers(
    short handle,
    Channel channel,
    short[] bufferMax,
    short[] bufferMin,
    int bufferLth);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetDataBuffer")]
public static extern short SetDataBuffer(
    short handle,
    Channel channel,
short[] buffer,
    int bufferLth);

[DllImport(_DRIVER_FILENAME, EntryPoint =
    "ps4000SetTriggerChannelDirections")]
public static extern short SetTriggerChannelDirections(
    short handle,
    ThresholdDirection channelA,
    ThresholdDirection channelB,
    ThresholdDirection channelC,
    ThresholdDirection channelD,
    ThresholdDirection ext,
    ThresholdDirection aux);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000GetTimebase")]
public static extern short GetTimebase(
    short handle,
    uint timebase,
    int noSamples,
    out int timeIntervalNanoseconds,
    short oversample,
    out int maxSamples,
    ushort segmentIndex);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000GetValues")]
public static extern short GetValues(
    short handle,
    uint startIndex,
    ref uint noOfSamples,
    uint downSampleRatio,
    DownSamplingMode downSampleDownSamplingMode,
    ushort segmentIndex,
    out short overflow);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetPulseWidthQualifier")]
public static extern short SetPulseWidthQualifier(
    short handle,
    PwqConditions[] conditions,
    short numConditions,
    ThresholdDirection direction,
    uint lower,
    uint upper,
PulseWidthType type);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetTriggerChannelProperties")]
public static extern short SetTriggerChannelProperties(
    short handle,
    TriggerChannelProperties[] channelProperties,
    short numChannelProperties,
    short auxOutputEnable,
    int autoTriggerMilliseconds);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetTriggerChannelConditions")]
public static extern short SetTriggerChannelConditions(
    short handle,
    TriggerConditions[] conditions,
    short numConditions);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetTriggerDelay")]
public static extern short SetTriggerDelay(short handle, uint delay);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000GetUnitInfo")]
public static extern short GetUnitInfo(short handle, StringBuilder infoString, short stringLength, out short requiredSize, int info);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000RunStreaming")]
public static extern short RunStreaming(
    short handle,
    ref uint sampleInterval,
    ReportedTimeUnits sampleIntervalTimeUnits,
    uint maxPreTriggerSamples,
    uint maxPostPreTriggerSamples,
    bool autoStop,
    uint downSamplingRation,
    uint overviewBufferSize);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000GetStreamingLatestValues")]
public static extern short GetStreamingLatestValues(
    short handle,
    ps4000StreamingReady lpps4000StreamingReady,
    IntPtr pVoid);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetNoOfCaptures")]
public static extern short SetNoOfRapidCaptures(
    short handle,
    ushort nWaveforms);
```csharp
[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000MemorySegments")]
public static extern short MemorySegments(
    short handle,
    ushort nSegments,
    out int nMaxSamples);

[DllImport(_DRIVER_FILENAME, EntryPoint = "ps4000SetDataBufferBulk")]
public static extern short SetDataBuffersRapid(
    short handle,
    Channel channel,
    short[] buffer,
    int bufferLth,
    ushort waveform);

[DllImport(_DRIVER_FILENAME,EntryPoint = "ps4000GetValuesBulk")]
public static extern short GetValuesRapid(
    short handle,
    ref uint noOfSamples,
    ushort fromSegmentIndex,
    ushort toSegmentIndex,
    short[] overflows);

#endregion
```
using System;
using System.Runtime.InteropServices;

namespace jPicoFishfinder
{
    public class PinnedArray<T>
    {
        GCHandle _pinnedHandle;
        private bool _disposed;

        public PinnedArray(int arraySize) : this(new T[arraySize]) { }

        public PinnedArray(T[] array)
        {
            _pinnedHandle = GCHandle.Alloc(array, GCHandleType.Pinned);
        }

        ~PinnedArray()
        {
            Dispose();
        }

        public T[] Target
        {
            get { return (T[])_pinnedHandle.Target; }
        }

        public static implicit operator T[](PinnedArray<T> a)
        {
            if (a == null)
                return null;
            else
                return (T[])a._pinnedHandle.Target;
        }

        public void Dispose()
        {
            if (!_disposed)
            {
                _pinnedHandle.Free();
                _disposed = true;
                GC.SuppressFinalize(this);
            }
        }
    }
}
Acknowledgements

Our activities are supported by “Coastal Area-capability Enhancement in Southeast Asia” Project, Research Institute for Humanity and Nature (RIHN, Project No.14200061).

We would like to thank Mr. Yuzo Abe of FUSION Inc. for their technical support and Dr. Kazuo Amakasu of Tokyo University of Marine Science and Technology for his theoretical advice. Thanks to Mr. Soichiro Matsushita (currently belonging to Honda Electronics Co., Ltd.) who vigorously conducted the experiment.

Research Institute for Humanity and Nature
February 2017

ISBN 978-4-906888-35-1