2950 Niles Road, St. Joseph, MI 49085-9659, USA
269.429.0300 fax 269.429.3852 hq@asabe.org www.asabe.org

CSBE | SCGAB

An ASABE – CSBE/ASABE Joint Meeting Presentation

Paper Number: 141929380

ISOBlue: An Open Source Project to Bring Agricultural Machinery Data into the Cloud

Alexander W. Layton[1], Andrew D. Balmos[1], Suppatach Sabpisal[1], Aaron Ault[1], James V. Krogmeier[1], Dennis Buckmaster[2]

[1]Electrical and Computer Engineering, Purdue University, 465 Northwestern Ave, West Lafayette, IN
[2]Agricultural and Biological Engineering, Purdue University, 225 S. University St, West Lafayette, IN

Written for presentation at the

2014 ASABE and CSBE/SCGAB Annual International Meeting

Sponsored by ASABE

Montreal, Quebec Canada

July 13 – 16, 2014

(The ASABE disclaimer is in a table which will print at the bottom of this page.)

Abstract. The past decade has seen a tremendous increase in the number of agricultural machines whose internal data communications follow the ISO 11783 (ISOBUS) standard. With this standardization in place, there is an excellent opportunity for data acquisition. In many cases, the data never leaves the vehicle; even when it does, it is not generally in a format useful to the farm management team and therefore is consequently under-utilized. Independent efforts to ameliorate the situation are hindered by an inability to get the ISOBUS data from the tractor into analytical tools. The goal of the ISOBlue project is to facilitate this data acquisition process through the development of an open-source hardware platform and software libraries to forward ISOBUS messages to the cloud.

This paper presents an overview of what ISOBlue is, how it is made, and how it works. A few of the ways ISOBlue has already been used to acquire and utilize data from an agricultural vehicle are shown. One use is forwarding data to an Android device to be visualized. Another use is forwarding data to an Android device to log and send the data to the cloud.

Keywords. *Precision agriculture, ISOBUS, data collection, mapping, network, technology, data, machinery, cloud computing, mobile applications.*

Introduction

Agriculture today utilizes numerous pieces of machinery. With more and more of this machinery using standardized ISOBUS data networks. This standardization means there is a growing opportunity to utilize real-time data from a variety of agricultural machinery. If such data could be delivered to where they are useful, farm management could be performed more effectively (Askey et al., 2013; Robert, 2002; Whelan & McBratney, 2001; Nikkilä, et al., 2010; Rains et al., 2011).

ISOBUS (ISO 11783)

ISOBUS is a communication protocol defined in the ISO 11783 standard (ISO, 2007a), and is used in the agricultural industry. The standard is based on the SAE J1939 standard. Most modern agricultural machinery is equipped with ISOBUS for its internal data communications.

ISOBUS Overview

An ISOBUS network is composed of two separate CAN busses, referred to as the tractor (or engine) bus and the implement bus. The bitrate of a network is 250 kbps (per bus). Data sent on an ISOBUS network have the format shown in figure 1 (ISO, 2007b). The data of a message is identified by its PGN (Parameter Group Number). To interpret a message's data, one must have access to the specification of its corresponding Parameter Group. The standard defines the data contained in many PGNs, but it also leaves many PGNs for proprietary use and does not define anything about the data corresponding to those PGNs.

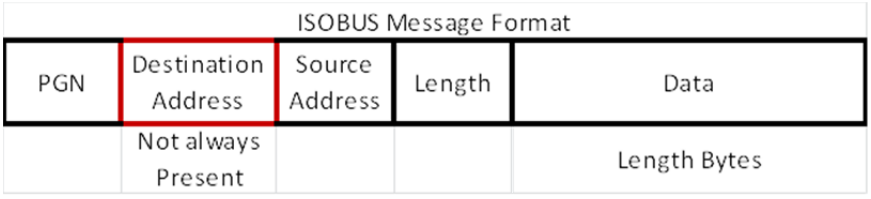

Figure 1. The format of messages sent over an ISOBUS network.

ISOBUS Data

A variety of useful data are present on the ISOBUS network of agricultural machinery (Steinberger et al., 2009). One example is "as applied" data about planting or applying treatments to a field (e.g., fertilizer). Some of what that data includes are location, speed, target application rate, actual application rate. Another example is yield data about harvest. Some of what that includes is mass flow rate, moisture, location, speed.

Objectives

The first objective of the ISOBlue project is to create an open source platform for accessing the data available on the ISOBUS network of enabled machinery. The second objective is to use that platform to get that data into the cloud where it can be used. The last objective is to create open source software to utilize ISOBUS data once it is somewhere accessible (such as the cloud).

ISOBlue

ISOBlue includes both hardware and software components. These components, along with an Android device running a third party app, enable accessing agricultural machinery data and bringing that data into the cloud. The various components of ISOBlue and examples of apps using ISOBlue are discussed in the following sections.

ISOBlue Device

The ISOBlue device is the hardware component of the system. The platform it is based on is a BeagleBone Black from the BeagleBoard.org Foundation (BeagleBoard.org, 2014). The device runs Angstrom, a kind of Linux, as its operating system. In addition to the BeagleBone, an ISOBlue device also includes a BeagleBone expansion board, generically called *a cape*, and a Bluetooth adapter. The ISOBlue device connects to an ISOBUS network with the CAN cape (TowerTech TT3201) which has multiple CAN interfaces. The CAN cape is compatible with SocketCAN, the de facto standard Linux driver for CAN. Using the USB Bluetooth dongle, the ISOBlue device can forward ISOBUS data over Bluetooth. Figure 2 shows an assembled ISOBlue device, and indicates where the parts are and how they connect. Figure 3 lists a set of parts when can be used to assemble an ISOBlue device.

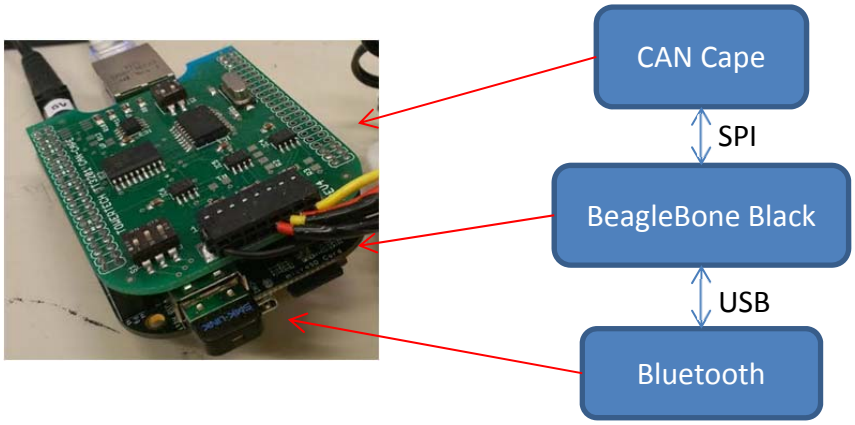

Figure 2. Components of the ISOBlue device and how they are connected.

Description	Source	Part Number	Price
BeagleBone Black	mouser.com	595-BB-BBLK-000	$45.00
CAN Cape	towertech.it	TT3201	79,00 €
Bluetooth Adapter	newegg.com	N82E16833412001	$22.99
MicroSD Card	newegg.com	N82E16820147236	$5.99

Figure 3. Parts for building a prototype ISOBlue device.

Building an ISOBlue

The process of assembling an ISOBlue device and loading the necessary software onto it is covered in Layton (2014b). A video which demonstrates that process, as well as using the created ISOBlue was created by Krogmeier and Layton (2014). In the video, an ISOBlue device is assembled and loaded with software, and then it is connected to a tractor. An Android tablet running the Logging/Cloud App (Layton, 2014a) connects to the ISOBlue, receiving ISOBUS data from it and forwarding those data to the cloud. Figure 3 lists the specific parts used in assembling the ISOBlue in the video, and some relevant information about those parts.

Software

The software of ISOBlue involves two main components: the *ISOBlue Daemon* which is software running on the ISOBlue device and the Android software library called *libISOBlue*. These components are discussed in the following sections. Figure 4 shows an overview of how data moves among components of the ISOBlue system. Portions of the figure will be discussed in detail in their related sections.

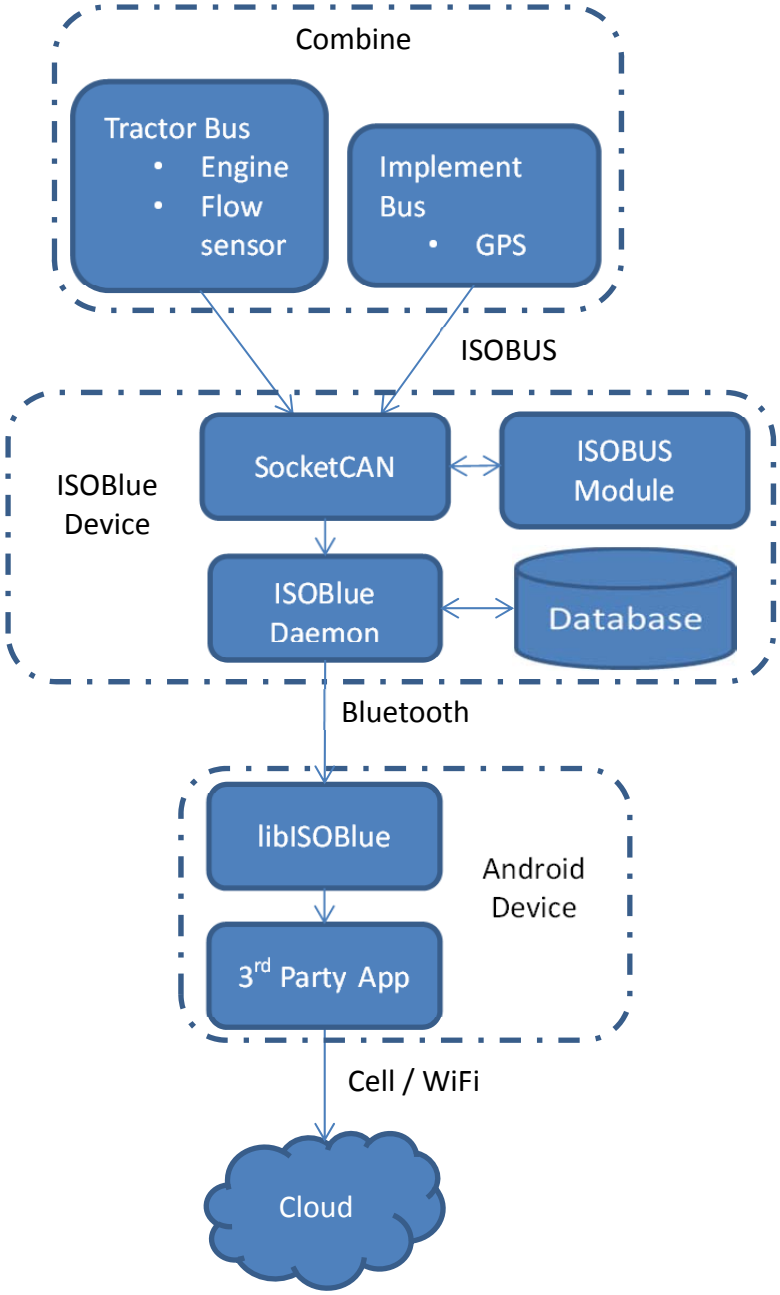

Figure 4. Data flow through the ISOBlue system.

ISOBlue Daemon Software

The ISOBlue Daemon is software written by Layton (2014c) to work as a "server" for the libISOBlue Android software library. It runs on an ISOBlue device, waiting for an Android device to connect using libISOBlue. The software concurrently monitors the ISOBUS network with SocketCAN and a custom written ISOBUS kernel module for SocketCAN. The ISOBUS kernel module is an addition to SocketCAN, which runs as part of Linux rather than as an application. The module adds ISOBUS protocol support to SocketCAN. Figure 5 shows that the ISOBUS module is a part of the data flow, but is not directly used by the ISOBlue Daemon.

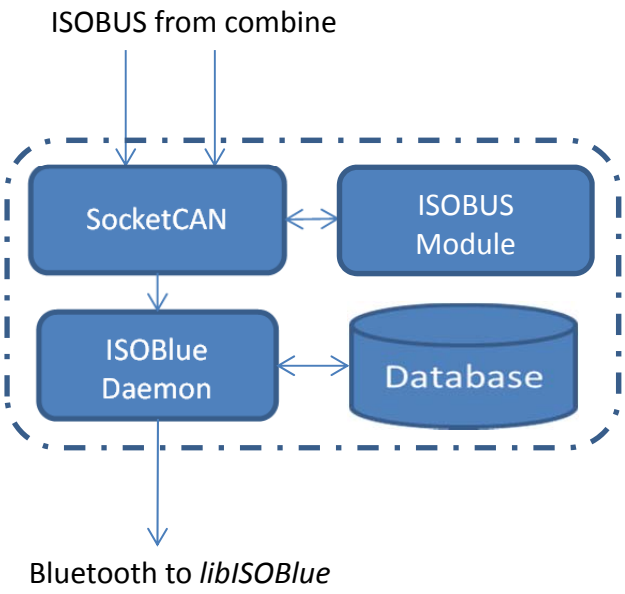

Figure 5. Data flow within the ISOBlue Daemon software.

The ISOBlue device always stores received messages in a local database on the BeagleBone's SD card. This is done in order to create a local log of all received data, and also to let libISOBlue on the Android device sync past data as needed. The database used is LevelDB from Google, which was chosen because it is fast and has low processing overhead. While an Android device is connected with libISOBlue, the daemon forwards ISOBUS messages from SocketCAN and/or the database to the Android device. Figure 5 demonstrates the flow of data. Data can go from SocketCAN to the daemon and out over the Bluetooth. Data can also go from SocketCAN to the daemon to the database, and later from the database back to the daemon and then out over the Bluetooth.

libISOBlue Android Library

The ISOBlue Android software library, called libISOBlue, was custom written by Layton (2014a) in Java to run on an Android device running Android version 2.3.3 or higher. The library receives ISOBUS messages forwarded from the ISOBlue device by the ISOBlue Daemon. It talks to the daemon over Bluetooth. The library tries to automatically reconnect when the connection to the ISOBlue device is lost. libISOBlue passes any forwarded messages on to the Android application which called the library. The flow of data within the Android device using libISOBlue is shown in figure 6.

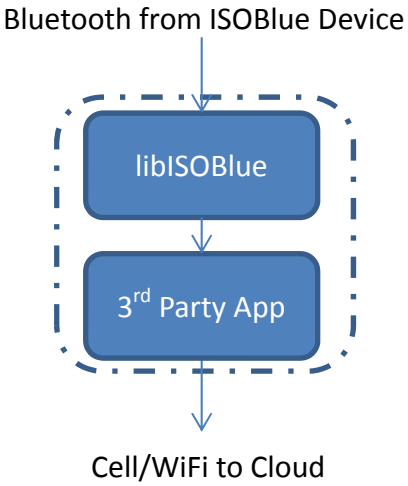

Figure 6. Data flow within the Android device using libISOBlue.

libISOBlue presents a simple socket like interface to the Android app using it. At a basic level a socket is an object which, when you read from it, you are returned a piece of data someone else sent to you since the last

time you read the socket. These sockets can be used to read data off either bus of the ISOBUS network to which the ISOBlue device is connected. In addition to the real-time sockets, buffered sockets can be used to read data from before the Android device connected to the ISOBlue device.

When read, the sockets return information about a single ISOBUS message (fig. 1). The information included is the PGN, source address, destination address, data bytes of the associated ISOBUS message, and a timestamp of when the ISOBlue device received the message over the ISOBUS network. The sockets can be set to filter out (not receive) certain messages based on their PGN. These filters can be configured separately on the two busses of the network.

Applications Using ISOBlue

Yield Monitor App

This Android app was written by Sabpisal (2014a). It receives GNSS and grain flow ISOBUS messages using the libISOBlue Android software library. The app uses the data contained in these received messages to generate a map of grain flow and shows that map on top of a Google Map. The map updates in real-time as ISOBUS messages are received. A demo video of this app is available from Sabpisal (2014b). Figure 7 shows a screenshot of the app generating a map of grain flow.

Figure 7. Screenshot of Yield Monitor Android App by Pat Sabpisal. This app uses libISOBlue to generate a real-time map of grain flow on top of a Google Map.

Logging/Cloud App

This app was written to test and demonstrate the capabilities of ISOBlue, and is included in Layton (2014a). It uses libISOBlue to receive all ISOBUS messages from the ISOBUS network. The messages are displayed on the Android screen as they are received, separated based on which bus they came over. The received messages are also logged to an SQLite database on the Android device's SD card and uploaded to the cloud (currently a spreadsheet in Google Drive). This app is shown working in the video by Krogmeier and Layton

(2014).

Future Applications

ISOBlue provides a low-cost platform and open libraries to facilitate ISOBUS data collection and interpretation.

ISOBlue can access and upload virtually any data on the CAN bus at a user-desired frequency. Applications include testing and monitoring, archiving data for farm information management, fault code identification, efficiency tracking, and more. Currently an Android device is needed, but future versions may be independent of a phone with an app.

For example, ISOBlue could map metrics of machine performance such as transmission settings, engine speed, engine load, and fuel consumption (Darr, 2012)). Such a map could be used to improve operational efficiencies.

Enabling ISOBlue to forward data directly to the cloud would facilitate writing cloud apps. Any of the previously mentioned Android apps could be implemented in the cloud, allowing managers and other off-site parties to monitor machine data in real-time. Cloud apps would even be able to integrate ISOBUS data from multiple ISOBlue equipped machines simultaneously, allowing for more possibilities of value.

Summary

Because of the ubiquity of ISOBUS networks on modern agricultural machinery, ISOBlue enables the effective use of agricultural machinery data for precision agriculture. These data and their potential uses are numerous and target users including manufacturers, fleet owners, rental firms, dealerships, and farm owners and operators.

Using commercially available parts and pre-existing or newly written open source software, an ISOBlue was created and used to forward ISOBUS data to an Android device. Examples of both visualizing this data and storing it in the cloud were demonstrated in this work.

This work showed that with the created system, it would be relatively simple to create cloud based and/or real time applications utilizing ISOBUS data from agricultural machinery. The implementation of more cloud based ISOBlue apps will greatly increase the usefulness of agricultural machinery data.

Acknowledgements

This work has been support by USDA/NIFA Project title: IMPROVING AGRICULTURAL MANAGEMENT WITH AUTOGENIC MOBILE TECHNOLOGY. It was additionally supported by the companies CrescoAg, Enterprise Group Ltd., Oxbow Agriculture LLC, and FarmLogs.

References

Askey, J., Darr, M. J., Convington, B., & Brue, J. (2013). Automated logistics processing of GIS data for agricultural harvest equipment. ASABE Paper No. 131596410. St. Joseph, Mich.: ASABE doi:10.13031/aim.20131596410

BeagleBoard.org (2014). BeagleBone Black. Retrieved from http://beagleboard.org/black.

Darr, M. (2012). CAN Bus technology enables advanced machinery management. doi: 10.13031/2013.42312. Resource 19(5): 10-11.

ISO (2007a). 11783-1: Tractors, machinery for agriculture and forestry - serial control and communication network Part 1: General standard for mobile data communication. Geneva, Switzerland: ISO.

ISO (2007b). 11783-3: Tractors, machinery for agriculture and forestry - serial control and communication network Part 3: Data link layer. Geneva, Switzerland: ISO.

Krogmeier, C., & Layton, A. W. (2014). ISOBlue setup: from zero to “in the cloud” in less than 4 minutes. Retrieved from http://youtu.be/Eq3BGirpObg.

Layton, A. W. (2014a). ISOBlue Android. ISOBlue Group. doi:10.5281/zenodo.10790

Layton, A. W. (2014b). ISOBlue Setup Tutorial. ISOBlue Group. Retrieved from http://git.io/iGhVvg.

Layton, A. W. (2014c). ISOBlue Software. ISOBlue Group. doi:10.5281/zenodo.10788

Nikkilä, R., Seilonen, I., & Koskinen, K. (2010). Software architecture for farm management information systems in precision agriculture. Computers and Electronics in Agriculture, 70(2), 328-336. doi:10.1016/j.compag.2009.08.013

Rains, G. C., Olson D. M., & Lewis, W. J. (2011). Redirecting technology to support sustainable farm management practices. Agricultural Systems, 104(4), 365-370. doi:10.1016/j.agsy.2010.12.008

Robert, P. C. (2002). Precision agriculture: a challenge for crop nutrition management. Plant and Soil, 247(1), 143–149.

Sabpisal, S. (2014a). ISOBlue Yield Monitor App. ISOBlue Group. doi:10.5281/zenodo.1081

Sabpisal, S. (2014b). ISOBlue Yield Monitor Demo. Retrieved from http://youtu.be/qLbVUeMaR_8.

Steinberger, G., Rothmund, M., & Auernhammer, H. (2009). Mobile farm equipment as a data source in an agricultural service architecture. Computers and Electronics in Agriculture, 65(2), 238-246. doi:10.1016/j.compag.2008.10.005

Whelan, B. M., & McBratney, A. B. (2001). The ‘‘null hypothesis’’ of precision agriculture management. Precision Agriculture, 2(3), 265-279.