

## **1. Overview**

### **RFID Inspection**

If you are producing RFID labels from label stock and inlays, you need to inspect your labels and verify that the inlays are still good after you convert them. The easiest thing to do is to run the finished labels past a reader, and check for the reader's "green light." That's slow, but it's the minimum investment path, and many companies start out that way.

### **Automated RFID Inspection**

As you begin to automate your testing, you first need to look at the steps involved:

- Monitor the web position so you know when to read
- Trigger the reader when the tag is in the read zone
- Complete some operations on the tag before it leaves the read zone
- Determine whether the tag was good or bad
- Mark the bad tag before it gets away

Each of those steps involves instrumentation that may or not already be on your system. Since it's your system, you will need to develop the exact sensor placement and reader setup commands for every tag you handle. You will also need to work out the exact timing sequences for your PLC programmers to implement. And if you don't own the PLC, you'll be negotiating with the equipment company for every change.

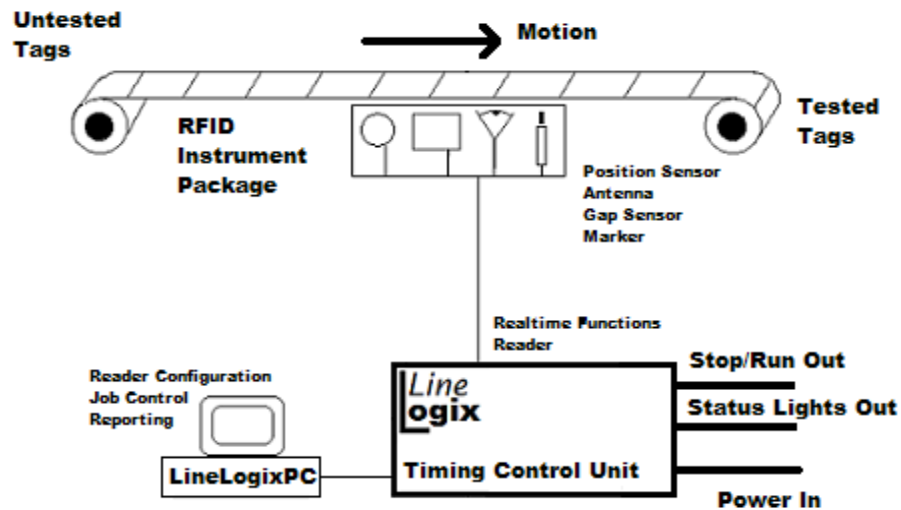
### **Interference and Overruns**

So, imagine you fight past all those issues and actually start running your system. What then? Well, first you find that you can't tell exactly which tag you're talking to. So now you're an antenna enclosure designer. Finally, everything is running great, and you start cranking the speed up, only to find that you are marking the wrong tags bad when the web exceeds some threshold speed.

### **LineLogix**

LineLogix is a standard platform for RFID conversion line automation. Using a combination of PC and realtime embedded software, it performs all the tasks outlined above, on different tag types, at high speed. It can tell when your line is running faster than your reader. The LineLogix Timing Control Unit (TCU) establishes the correct timing windows for reading and marking based on gap and position sensors. The TCU remembers how far your marker is from your reader, so the correct tag is marked, every time. LineLogixPC software configures it all, monitors the job, and reports the results.

## 2. System Diagram



## 3. Functionality

The LineLogix TCU monitors a Gap Sensor or PLC input to develop timing windows for reader and marker during forward web motion. Reading and marking are triggered by the leading edge of each label as it enters the Gap Sensor. There are no offsets for time or web position, except those inherent in system delays.

When a label is positioned such that its tag is in position for testing, the LineLogixTCU sends commands to an RFID reader to test the tag. Tests can be read-only or write-verify, and can be short screening tests or long tag write operations, depending on the application.

When the RFID reader's test is complete, the reader must provide a Good or Bad status reply to the LineLogix system. When the reader indicates that a tag was bad, that information goes to the TCU's marking logic. Depending on the location of the marker relative to the read point, configured by LineLogixPC, the bad tag can be marked immediately or when it gets to the mark point.

## 4. Tag Testing

At user's option, each reader in the system can perform one of the tests below.

- Count, i.e. Read with repeats within a window set by the Trigger input. Count Labels, saving distinct tag IDs if tags are already serialized.
- Read once per trigger (with or without ID logging)
- Write/Verify Constant or Incrementing Pattern, once per trigger

Tests can easily be added, or existing tests customized. Contact GlueLogix.

## 5. Performance

The LineLogix system can read inlays up to 20 per second (20/sec) with Feig and ThingMagic serial connected RFID readers. That means that the system can respond to triggers, send serial commands and parse replies that fast.

Few applications actually run that fast. For reasons from inlay sensitivity to operator skill, it is rare for the system to run at top speed. Most installations run at 10-15 per second as a top speed.

Performance cannot be stated in Feet Per Minute (FPM) without also stating the repeat. This table gives the FPM at various tag rates for common repeats:

	10 per second	15 per second	20 per second
1 inch repeat	50 FPM	75	100
2 inch repeat	100	150	200
4 inch repeat	200	300	400
6 inch repeat	300	450	600

As of 2018, the standard LineLogix antenna is physically small to support the widest possible variety of applications. The performance of long pitch applications like luggage tags can be improved by custom antennas that engage the inlay for several inches.

For more information on timing, request a spreadsheet from GlueLogix.

## 6. Risks

Because every LineLogix installation differs by tag type and customer needs, system performance must be validated experimentally after installation. See the QA and Job Setup sections of the LineLogixPC Reference manual for more information.

## 7. LineLogixPC Reference

Since LineLogixPC is sometimes sold separately, that reference is in a separate document. The following sections explore topics in which the PC software interacts with the TCU.

### 7.1. *LineLogixPC Adjustments*

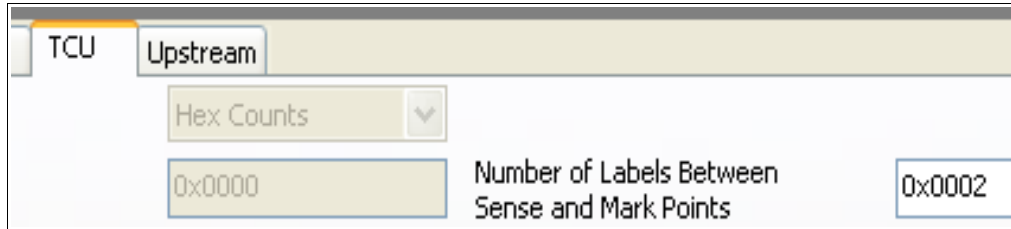
Once set up, a LineLogix job should only require small adjustments as new inlay types and label sizes are added to a shop's repertoire.

#### 7.1.1. Mark Position

LineLogix systems have the ability to integrate a large number of devices for marking

bad tags. When used on a card line, the Mark output could also trigger a diverter.

Systems so equipped must be configured with the number of labels between the antenna and the marker or other actuator:



The screenshot shows a software window with a tab labeled 'TCU' and a sub-tab labeled 'Upstream'. Below the sub-tab is a dropdown menu labeled 'Hex Counts' with a downward arrow. Underneath the dropdown are two input fields. The first field contains the text '0x0000'. To its right is the text 'Number of Labels Between Sense and Mark Points'. To the right of this text is a second input field containing the text '0x0002'.

In the example, the number 2 is determined by positioning the web at a transition from gap to paper, and counting the number of labels between the antenna and the marker. Do not include the label directly over the antenna, but do include the label under the marker.

Other LineLogix PC settings may be used to stop the web for operator recovery or replacement of bad tags. This feature is called Stop On Mark and is set in the TCU tab. The StopOnMark feature can be used with downstream marking to stop a bad label over a rework surface.

## **7.2. Quality Assurance**

Although LineLogix is a highly capable system, incorrect setup can result in incorrect results. Quality Assurance is the most important thing you can do to validate your setup.

With encoding, especially, it is very possible to get OK indications on a setup that is not what the end user needs.

QA guidelines are given in the LineLogixPC Reference.

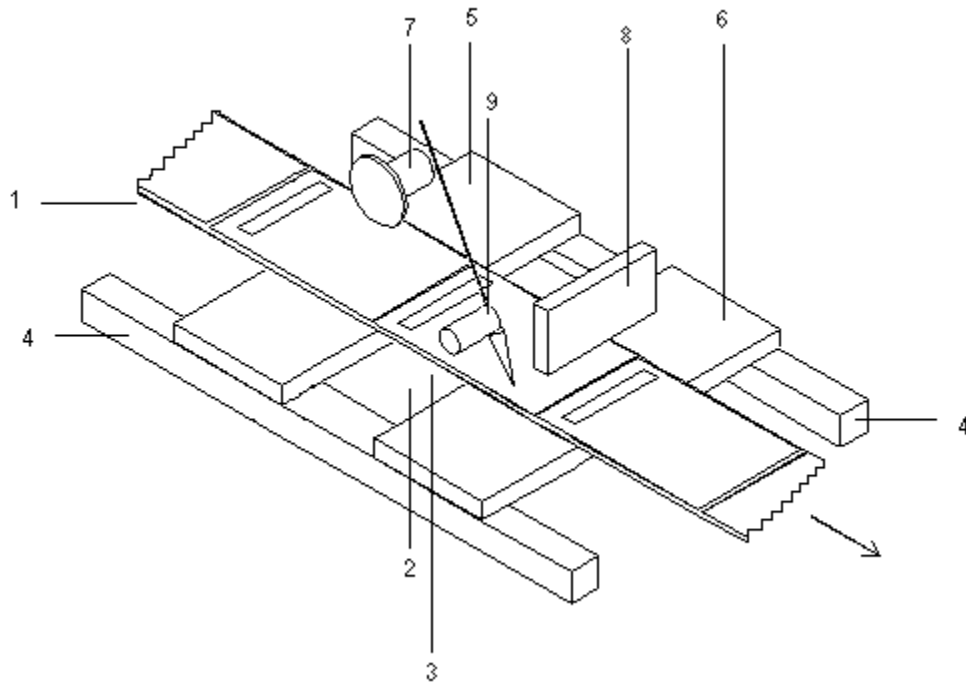
## **8. LineLogixRIP Reference**

The RFID Instrument Package (RIP) of the LineLogix system contains the RFID antenna and all sensors needed to test tags. It may also physically support a marking device.

### **8.1. Elements**

Several complementary views of the RIP are presented here because an understanding of the RIP elements is basic to an understanding of the LineLogix system.

### 8.1.1. Isometric Diagram

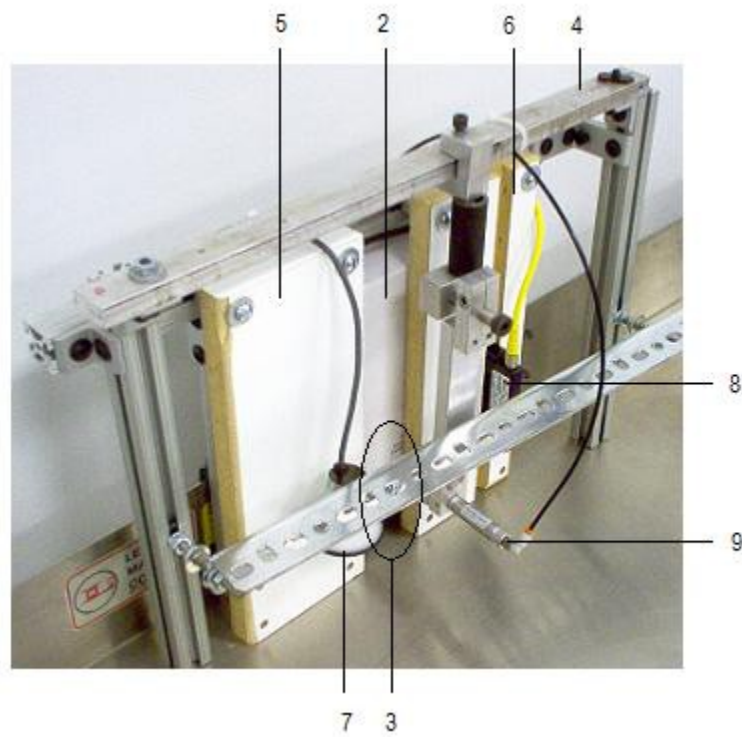


This drawing is from US Patent 7,375,636, Apparatus and method for real time functional testing of RFID tags, which protected the LineLogix design from unauthorized use through the year 2020. While the design has evolved, this diagram is still a good illustration of the concepts.

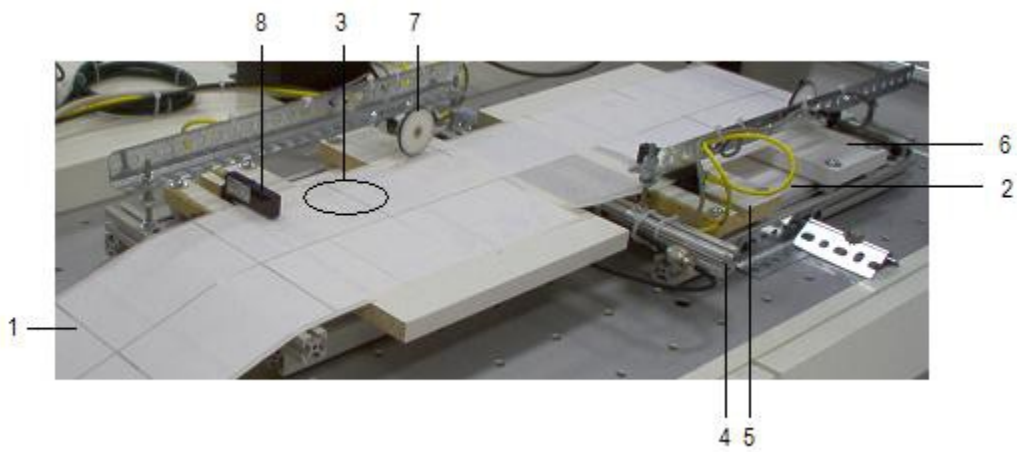
### 8.1.2. List of Numbered Parts

1. Continuous Web of RFID Tags
2. Test Antenna (below RF Baffles)
3. Test Section (portion of web exposed to antenna)
4. Support Brackets
5. Upstream RF Shield (called Baffle in the patent)
6. Downstream RF Shield
7. Shaft (Position) Encoder, unused since 2007
8. Gap Sensor
9. Marking Device

### 8.1.3. Early One-Up System on Iederle Rewind



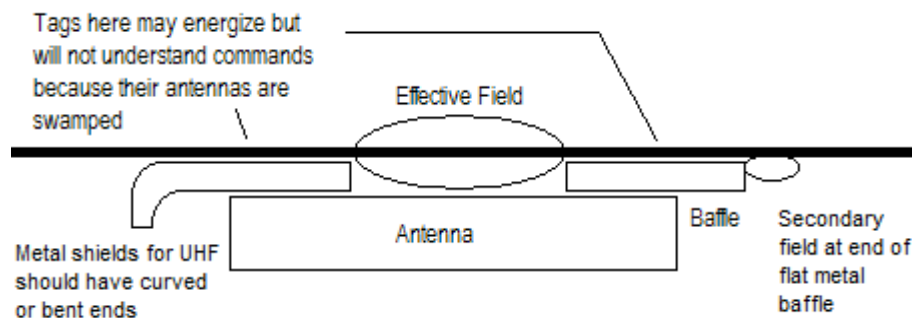
### 8.1.4. Early Two-Up System on ABG Press



## 8.2. RF Shielding

The RIP's RF shields are metal plates. The shields both direct the RF field and “swamp” the antennas of adjacent RFID tags outside the Test Section by changing the impedance and resonant frequency of those antennas.

Shielding can be any metal that is at least 0.010 inch (ten thousandths) thick. Common roof flashing works very well. Aluminum foil is too thin. 0.002 inch thick copper tape works because of copper's high conduction, but sticky tape is too hard to adjust for variable spacing.

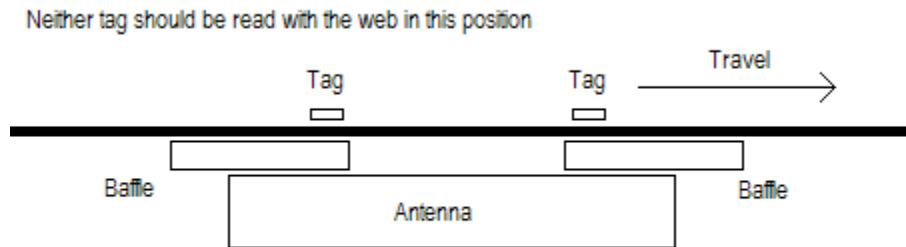


RFID tags MUST touch the shield material or be at most 1/16 inch away from the surface of the RF shields.

### 8.2.1. Setting Up RF Shielding

The RF baffles control tag singulation in the LineLogix RIP. The opening between RF Baffles is known as the Test Section. Tag singulation improves with a smaller Test Section, but throughput improves with a larger one.

1. Loosen the holddown screws.
2. Slide the shields so that the distance between them is the same as your label length.
3. Slide both shields together so that the gap between them is centered over the antenna.
4. Position the web so that one tag is on the upstream RF baffle, just about to enter the test section.



5. Read tags to make sure that no tags are visible. This LineLogixPC button definition will work for most installations once a job is loaded:

```
Read Once, self.OnReadOnce()
```

6. If tags are read in this condition, reduce reader power or close up the gap until no tags are read.
7. Move the web so that the tag travels across the test section. Ensure that one and only one tag is read during the entire transition. If several tags fail to read consistently as they travel across the test section, raise power a little.
8. Move the web so that the tag is on the downstream shield. Ensure that no tag is read with the web in this position.
9. If multiple tags are read at any step, lower reader modulation or power or close up the baffles. In some UHF applications, it may be necessary to bend down the free ends of the shields against conduction of power into upstream or downstream inlays.

If you cannot singulate tags using this method, contact GlueLogix. Take a photo of the setup and email it to GlueLogix.

To optimize the setup, spread the shields further apart and repeat the previous steps until you find the maximum width for your application. This will allow you to increase the web speed for the application to the maximum extent possible for reliable operation.

When you are satisfied with the shield position, document it for future runs of the same test.

### **8.3. Near Field Antennas**

GlueLogix has a number of high quality near field antennas suitable for tag testing.

UHF near field antennas may be built as point or line devices. Point antennas are meant to engage one inlay. Line antennas may be placed across the web to engage one inlay, or along the web to engage multiple inlays (or one inlay for a longer time as with luggage tags).

HF antennas may be built as circuit boards or coaxial loops. Circuit board antennas, especially the Feig 40x30mm device, are generally meant for use as point antennas (one inlay). Coaxial loops are wound to be installed along the web and engage multiple



inlays.

In GlueLogix USA patent 8,896,425, Apparatus and method for serialized continuous encoding of RFID tags, the terms “short” and “long” are introduced in describing near field antennas for RFID work.

- Point antennas and strips laid across the web to engage single inlays are called “short” antennas.
- Strip and loop antennas laid along the web to engage multiple inlays are called “long” antennas.

### 8.3.1. Point Antennas

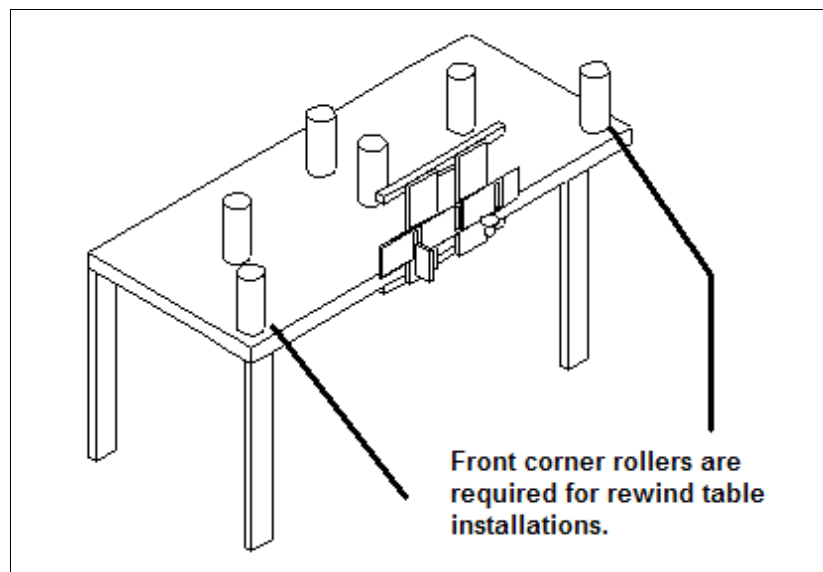
Because point antennas have narrow fields, they are mainly used for read testing (Read and Count tests) in on-press applications. They are only useful for writing (Write and Serialize tests) if the transport mechanism is able to stop the web with the inlay precisely over the antenna, as in rewind applications.

Point antennas are required for close pitch applications.

### 8.3.2. Rewind Table Installation with Near Field Antennas

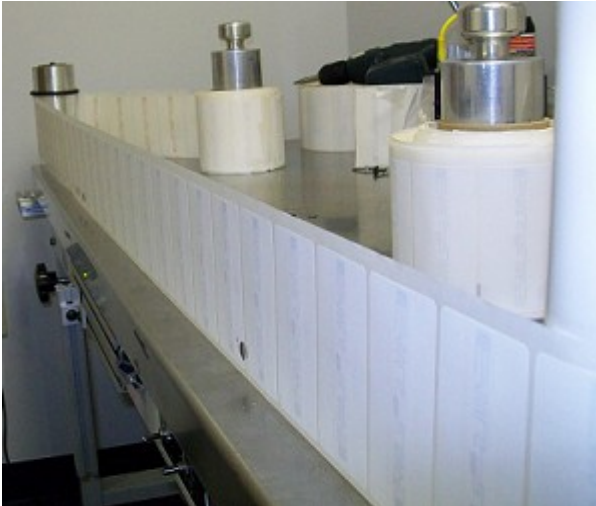
This section shows the installation of a near field RIP on a rewind table fitted with front corner rollers. Corner rollers allow the web to travel past the antenna at a constant angle as the supply and takeup rolls change size. Front corner rollers allow the antenna assembly to sit at the front edge of the table for easy threading and adjustment.

1. This picture shows the placement of front corner rollers:



2. Once the front corner rollers are affixed and the table is threaded, the paper web will

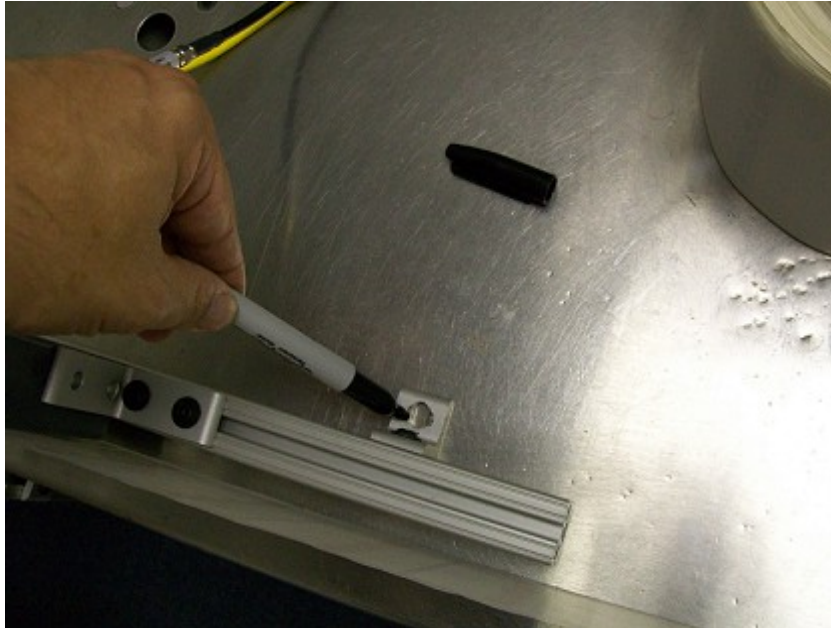
have a natural path across the front of the table, as shown here:



3. Assemble the gap sensor and antenna onto one piece of 80-20 to ensure alignment. Stand the assembly up on the table so that the web goes through the gap sensor and brushes the front of the antenna without deflecting:



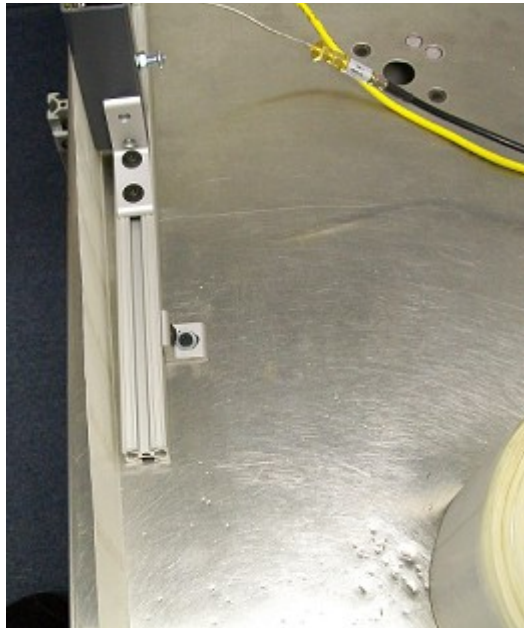
4. Mark the location for pilot holes. Move the mounting corners if necessary to avoid obstacles:



5. Drill pilot holes for wood screws that will grab the rewind table's wooden top under the steel shell:



6. Secure the RIP with wood screws and washers:

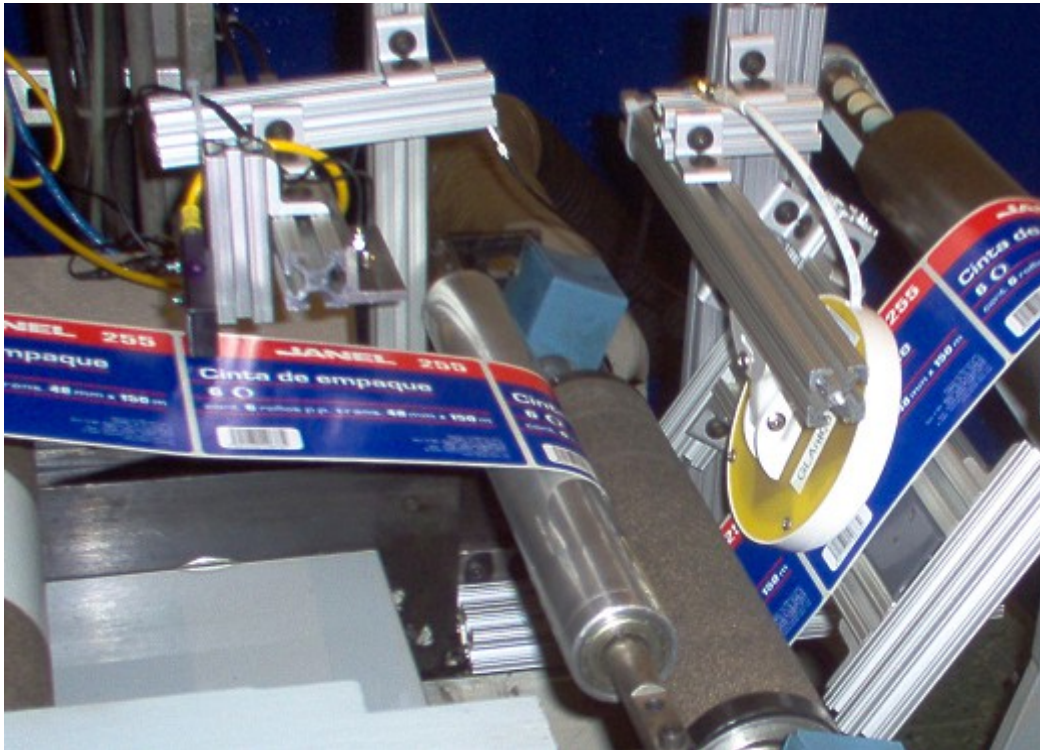


7. The finished installation should look like this.  
The dark bron device connected to the yellow cable is the Gap Sensor (8) above.  
The gray square tube is a short UHF antenna of 2010 design.



#### **8.4. Mixed Field Antenna Assemblies**

Some applications have worked best with a combination of near and far field antennas, arranged in a “sandwich” with the web between them. Details of these configurations are TBD. This photo shows such an installation:



In this setup, paper moves to the right. The Gap Sensor (8) is on the left, just upstream of the first visible gap. The Gap Sensor had to be made movable with an elaborate 8020 bracket, because the antenna position was fixed by rollers. The sandwich configuration made up for reflections from rollers and other press components, made necessary by the tight spacing.



## 8.5. Shaft (Position) Encoder

Note: The Shaft Encoder was a standard feature of early LineLogix systems with SAMSys readers. After the encoder proved too difficult for many customers to set up, it was abandoned in favor of a movable gap sensor; however, its settings are still supported in LineLogixPC and can be reintroduced to the TCU firmware if an application requires it.

The Shaft Encoder provided relative position and direction information to the TCU. It gets power from the TCU and provides two square wave signals to the TCU. Each full cycle of either signal is one “count” of web travel. The frequency of the signals indicates the speed of the web. Since the signals are 90 degrees out of phase, their phase relationship indicates the direction of web travel.

### 8.5.1. Translating Shaft Encoder Counts to Distance

The original RIP used a 100-count-per-revolution encoder with a drive wheel of 2.125 inch diameter and 6.676 inch circumference. Therefore, each Shaft Encoder count translates to 0.067 inches of web travel, and every 15 counts was almost exactly an inch.

<i>Counts</i>	<i>Distance</i>
4	0.267 inch
15 (0x0F)	1.00 inch
30 (0x1E)	2.00 inches
45 (0x2D)	3.00 inches
60 (0x3C)	4.00 inches

### 8.5.2. Setting Up the Shaft Encoder

The mechanical adjustment of the shaft encoder was the most sensitive adjustment in the LineLogix system. With practice it was easy to perform; however, a careless or incorrect adjustment could ruin labels.

- Place a label on the RF Baffle below the Shaft Encoder drive wheel.
- Loosen the adjustment screw and position the drive wheel so it is barely touching the label. Tighten the adjustment screw.
- Move the label smoothly in the normal direction of travel. The drive wheel should track the motion of the label.
- Move the label across the normal direction of travel (sideways). There should be no resistance to sideways travel.
- Repeat the above steps until the label can move sideways with no resistance, but

the drive wheel tracks normal motion.

If, during operation, the web is forced to one side or the other, it is likely that the drive wheel is too tight. Repeat the above steps and double check the tightness of the adjustment.

### 8.5.3. Troubleshooting the Shaft Encoder

Two red LEDs are provided near the Shaft Encoder connector. Observe those LEDs while slowly turning the Shaft Encoder drive wheel. They should cycle per this sequence:

<i>Phase A</i>	<i>Phase B</i>
Off	Off
On	Off
On	On
Off	On

Any deviation suggests a wiring problem or a bad sensor.

If the web is being pushed to one side in the RIP, the most likely explanation is that the Shaft Encoder drive wheel is pressing too hard on the paper. See the section above on adjustment.

## 8.6. Gap Sensor

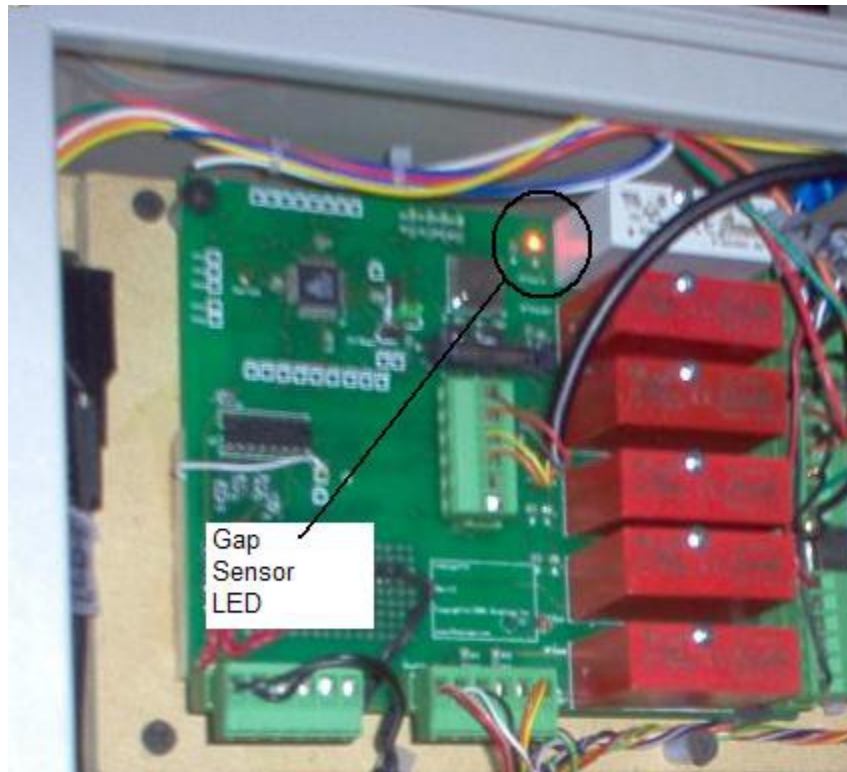
The Gap Sensor is an optical sensor that detects the translucent region between labels. If your application does not have visible gaps between labels, the Gap Sensor can be retrained to detect index marks; however, the system will not work unless there is one and only one index mark for each label. If these conditions cannot be met, contact GlueLogix for application support.

The rest of this section describes normal operation with the standard gap sensor, the Allen-Bradley LPT45. If your application uses a different gap sensor, the instructions for that device should have been provided to you by GlueLogix.

### 8.6.1. Checking the Gap Sensor

Position at least two labels on the RIP such that one label is in the Gap Sensor.

Make sure the Gap Sensor is dark, and that the TCUv3 LED associated with the Gap Sensor is also dark. This is D5, near the white Grayhill I/O module (U4) nearest the bottom edge of the TCU circuit board.



Slide the labels so that the gap is in the Gap Sensor.

Make sure the Gap Sensor's Green LED is on, and that the TCU's Gap Sensor LED is also on.

Slide the labels so that the next label is in the Gap Sensor.

Make sure the Gap Sensor is dark again, along with the TCU's Gap Sensor LED.

On TCUv4, the LED is D20, also in the top left corner.

### **8.6.2. Setting Up the Gap Sensor**

To set up the standard 45LPT Gap Sensor for opaque labels on a translucent web:

Position the web (or at least two labels) so that the translucent gap is inside the Gap Sensor.

Press and release the button on the top of the Gap Sensor. This is the “Teach Button.”

Verify that the Red LED inside the Gap Sensor is blinking. If not, wait a few seconds and repeat the last step with a slightly longer push.

While the Red LED is blinking, slide the labels back and forth quickly so that the Gap Sensor sees both opaque and translucent areas several times. Continue until the Red LED stops blinking.

Slide the material back and forth a few more times to make sure the Gap Sensor triggers where you want it to. Repeat teaching if necessary. Some materials may require contact



with the thicker fork of the 45LPT before correct training occurs.

Perform the test procedure above.

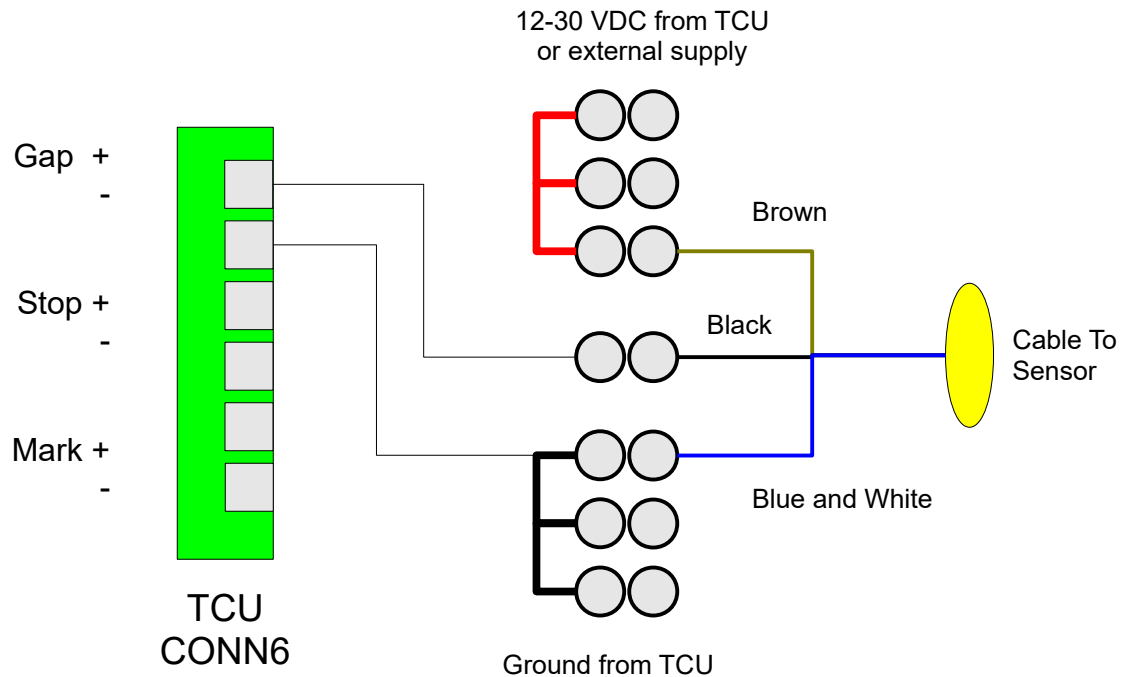
See also:

<http://gluelogix.com/Downloads/45LptTraining.mp4>

[http://gluelogix.com/Downloads/45lpt-in001\\_-en-p.pdf](http://gluelogix.com/Downloads/45lpt-in001_-en-p.pdf)

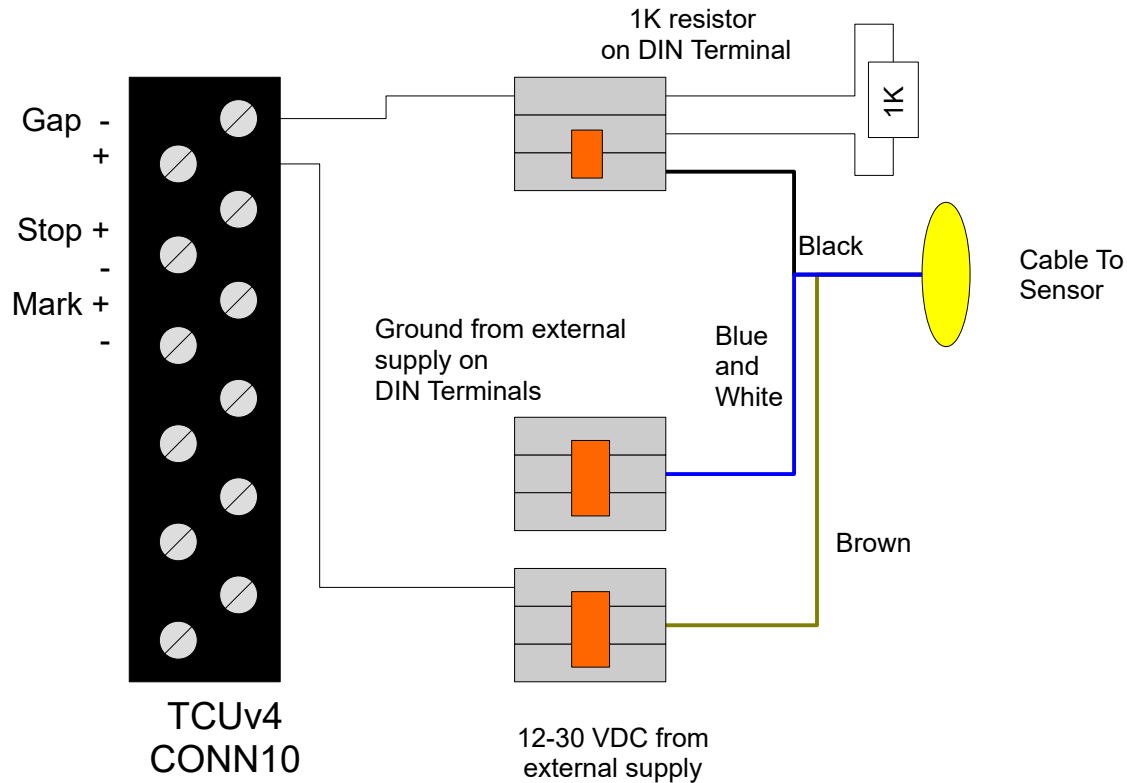
### 8.6.3. Installing the Gap Sensor on TCUv3

The Allen-Bradley 45 LPT and FVL sensors are both wired to LineLogix as shown. Color coding refers to the standard Allen-Bradley cable. This sensor requires a white DC Input Grayhill module in top position.



Other gap sensors can be integrated using the diagram above as a model. Be sure to select the correct Grayhill module to complete the input circuit.

#### 8.6.4. Installing the Gap Sensor on TCUv4



#### 8.6.5. Other Gap Sensor Information

To train the 45LPT Gap Sensor for translucent material with opaque index marks, push the Teach Button twice.

- The Gap Sensor has a lockout to prevent accidental retraining. If the Teach Button is locked, the Red and Green LEDs on the Gap Sensor itself (not the TCU board) will both be on steady.
- Push the Teach Button for six seconds to lock or unlock the Teach Button for normal operation.

The Allen Bradley (Rockwell Automation) 45FVL Fiber Optic Item Sensor is plug compatible with the 45LPT and may be substituted when the application requires it. The Fiber Optic may be set up as a through-beam or reflective sensor. The recommended configuration for applications with a black bar printed across the label gap is reflective, where the black bar reflects less light than either the unprinted label or the inlay region.

#### 8.6.6. Setting the Tamarack Read Window

The following procedure assumes a synchro press capable of very small jog movements.

- With inlays being fed by the Tamarack unit, stop the press.
- Start LineLogixPC, select an appropriate job, and click Start Roll to set up the TCU.

- Click Read Once to read a tag. In two-up jobs, click Read Gear to read on the gear side.
- Observe the Feedback tab for tag reads.
- Jog the press until no tag can be read.
- Jog the press until a tag just starts reading.
- On the Tamarack interface main screen, press Setup, Read Window, and then Advanced.
- Note the rotational position of the Tamarack system. Note, Tamarack rotates through 360 degrees on each ticket.
- Jog the press until that tag no longer reads using LineLogix debug buttons.
- Note the rotational position of the Tamarack system.
- Set the Tamarack Read Trigger to a value about 25% between the two rotational positions. That is, if the two positions are 0 and 100 degrees, set the Read Trigger for 25 degrees.
- Set the Tamarack Read Window for 3 inches, or 75% of the ticket length, whichever is less.

## 8.7. Marking Device

Marking Devices are highly application specific. They may be provided by the customer and easily integrated through the TCU's I/O Module mechanism at installation time by GlueLogix.

If a Marking Device is provided by GlueLogix, it will most likely be an HP inkjet print head made by Digital Design.

Many markers use **compressed air**. The customer is responsible for maintaining the air supply to the marker in a safe condition.

Marking ink is sometimes classified as a **hazardous material**. The customer **MUST** read, understand and follow all precautions necessary for their chosen marking ink, as presented by the supplier of that ink.

### 8.7.1. DDI Evolution Setup

The DDI Evolution comes with a mounting kit and complete instructions.

Make the following settings to a new Evolution printer:

- External Product Detector must be enabled (F2-3-2-Enter) unless the printer has been modified by GlueLogix. If the printer has a GlueLogix sticker indicating application of DDI Field Changes for noise immunity, the Internal Product Detect can be enabled for a longer debounce.
- Internal Encoder (F1-2-1-Enter) should be set by default, but should also be checked.
- If Quick Dry Ink is used, you must enter the special commands supplied with the ink whenever power is applied to the printer.

- Message is a single character. The printer comes with a long message that will spoil multiple labels in most configurations.

With these settings, an Evolution printer in Print mode can be tested by shorting pins 8 and 9 of the DDI's secondary port.

Note: The DDI Evolution communication ports are 10 pin versions of the more common 8-pin network jacks. CAT5 network jumpers may be used in these jacks. If so used, pin 1 of the CAT5 cable connects to pin 2 of the Evolution system. Evolution pins 1 and 10 are not connected. The external product detector is therefore triggered by shorting the last two wires of the CAT5 jumper, usually Brown and Brown/White. These same wires may be attached to a DC Output Module (Grayhill brown SSR) on the TCU Mark Output.

Make the following settings to LineLogixPC:

In LineLogixPC or the job file, set the mark interval to 0x0100 (256) and clear the InvertMark setting. If Internal Product Detect is selected on a modified printer, set the mark interval to 0x800 (2048).

See also:

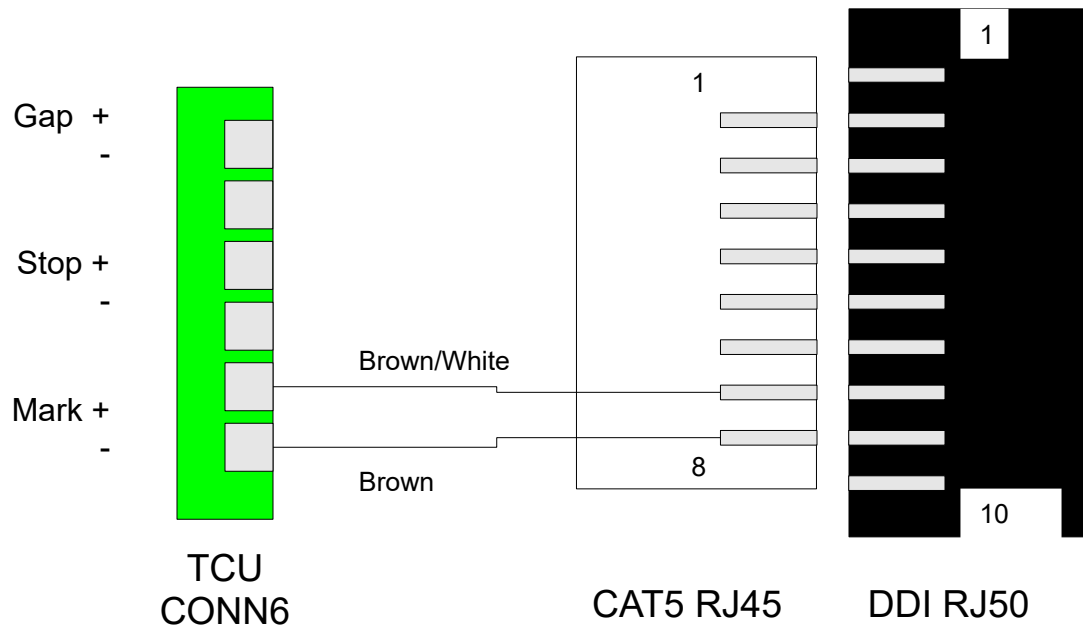
[http://gluelogix.com/Downloads/EV1MANUAL1\\_5.pdf](http://gluelogix.com/Downloads/EV1MANUAL1_5.pdf)

### 8.7.2. DDI Evolution Installation

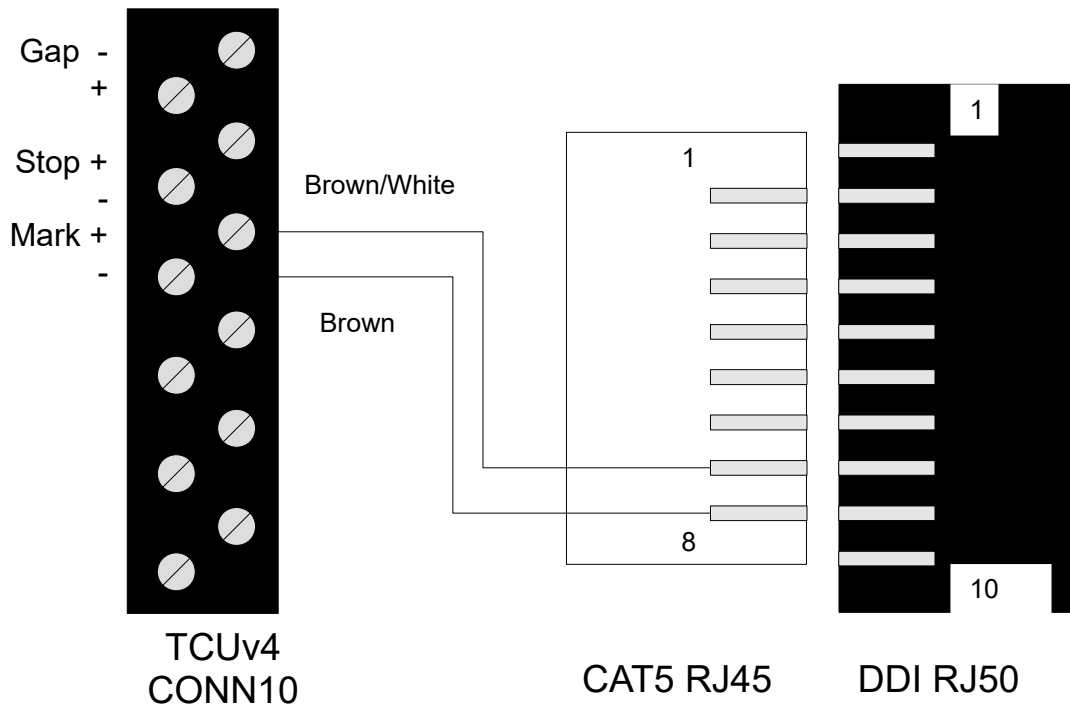
To wire the DDI Evolution into the TCU:

- Cut the end off a standard CAT5 network cable (if you have to use a swap cable, disregard the color code given below).
- Feed it into the TCU enclosure through a gland
- Strip the ends of the Brown and Brown/White conductors.
- Wire as shown and check with an ohmmeter.
- Plug into the secondary port of the DDI.
- Fit a DC Output (brown) Grayhill SSR into Mark socket U6 of the TCU board.
- **GROUND THE PRINTER BODY**

### 8.7.3. TCUv3 Connections



### 8.7.4. TCUv4 Connections



### 8.7.5. DDI Evolution Maintenance

For all inks:

- Do not reapply the tape to the nozzles of a print cartridge
- Clean the nozzles only with a dry lint-free wipe. Do not use water or solvents.

For fast dry inks, prevent dryout by:

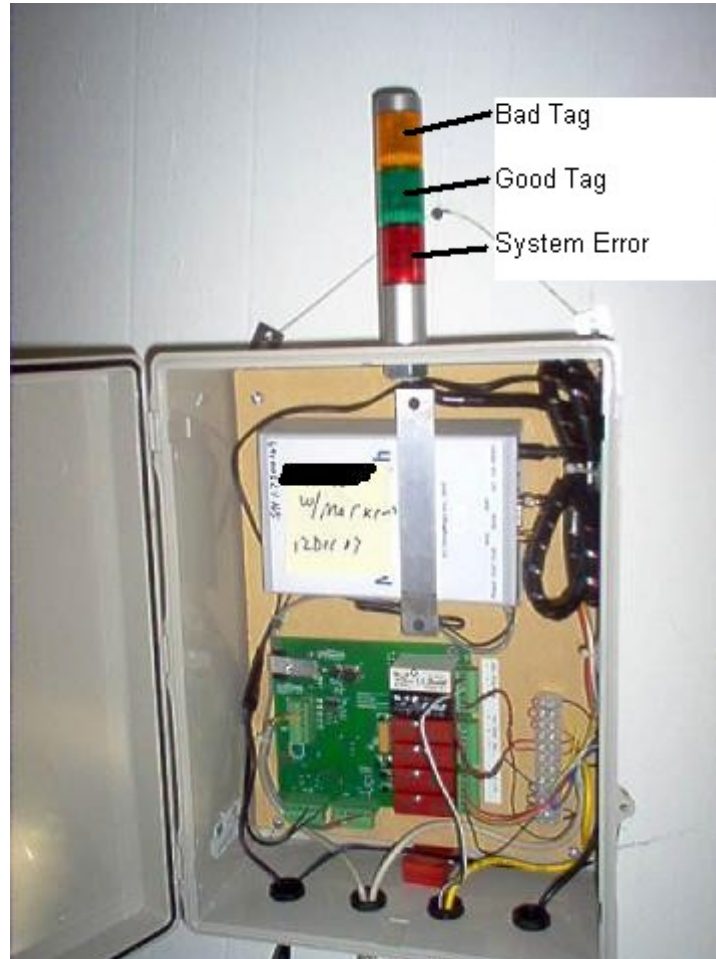
- When the printer is secured overnight, **always** remove the ink cartridge, replace it in its storage cradle, and replace both in the original ziplock plastic bag. Seal the bag.
- When the printer is turned back on at the start of the next shift, **always** execute the setup commands included with the print cartridge

### 8.7.6. DDI Evolution Troubleshooting

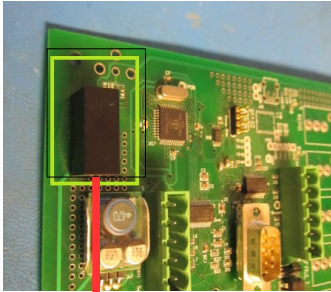
1. If there is a lot of electrical noise on the trigger line, the DDI will trigger randomly. DDI has issued a Field Change notice to combat this problem, which GlueLogix now applies to all imprinters we sell. If this is the problem, you will see the green LED flash, and ink come out, at unpredictable intervals. That behavior will stop if you unplug the trigger cable.
2. If a poorly grounded printer has been damaged by ESD, the green LED may blink constantly and the handheld controller may not be able to communicate with the printer body. That is a repair issue, contact GlueLogix to arrange.
3. If the ink cartridges are not properly maintained, they can dry out or sustain other damage. See the Maintenance notes above. If this is the problem, you will not see the green LED blinking under normal circumstances. You will be able to make it blink from the PC by choosing Mark 0 or Debug -> M00! in the TCU menu. If that works as described but no ink comes out, your ink cartridge is dried out or damaged.
4. At least one Evolution 1 printer has lost fonts after a long shutdown. GlueLogix keeps a special memory card that can be sent to trouble sites to fix this problem.

### 8.8. *Light Pole*

The TCU uses a Light Pole for operator feedback. It is normally attached to the top of the TCU case, but can be removed and mounted remotely. The indicators are:



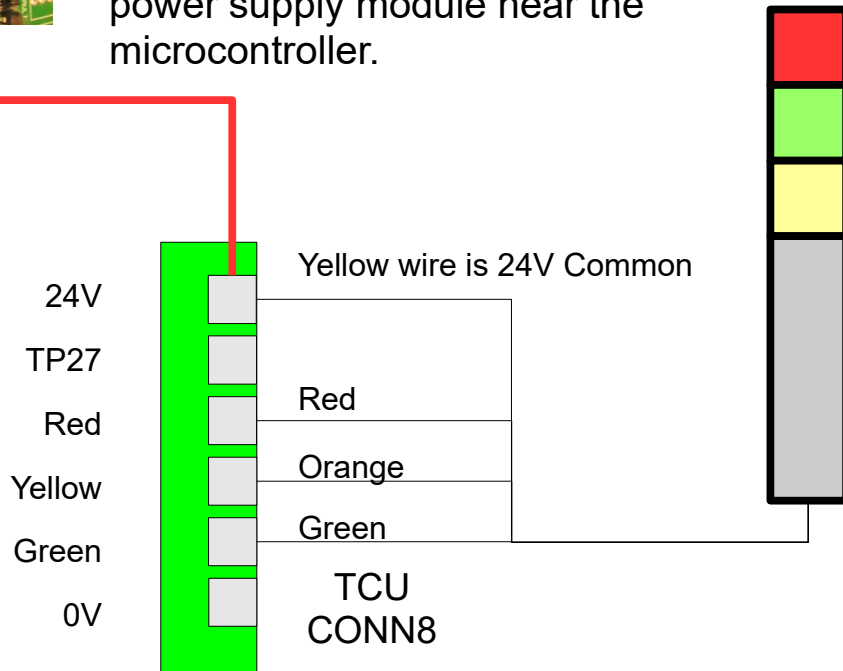
Some stack lights may have the colors in a different order, but the colors always mean what this picture says.



## LineLogix Systems with ThingMagic Readers

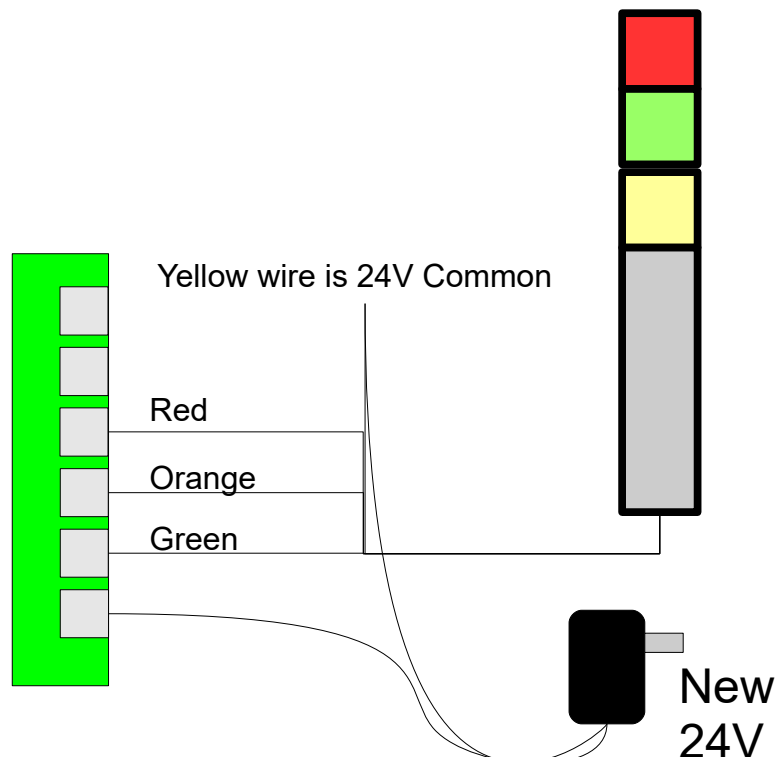
run from the reader's 12V supply and make 24V on board in an undesignated power supply module near the microcontroller.

To energize a light, the TCU switches 0V into the wire for the individual color light. In most delivered TCU V2 and V3, this switching is done by the transistor network to the left of the Grayhill I/O Module bank.



A TCU board with failed 24V circuit should be repaired or replaced. As a temporary workaround, a 24V power supply may be installed between TCU 0V and the yellow wire of the stack light.

**DO NOT CONNECT EXTERNAL 24V TO THE TCU BOARD.**





## 9. LineLogixTCU Reference

TCU Revision 1.x used a Parallax SX52 processor. TCU Revision 2.x used a Parallax SX48 microprocessor. Both chips were essentially a pipelined PIC style design, using a 50 MHZ clock to process system events at speeds approaching that of programmable logic.

TCU Revision 3.x and 4.x use a Parallax Propeller running at 80 MHZ for similar speed with more memory and better capacity for computations. The Propeller is an 8 core device, in which each core is a 32 bit RISC machine.

### 9.1. General Algorithm

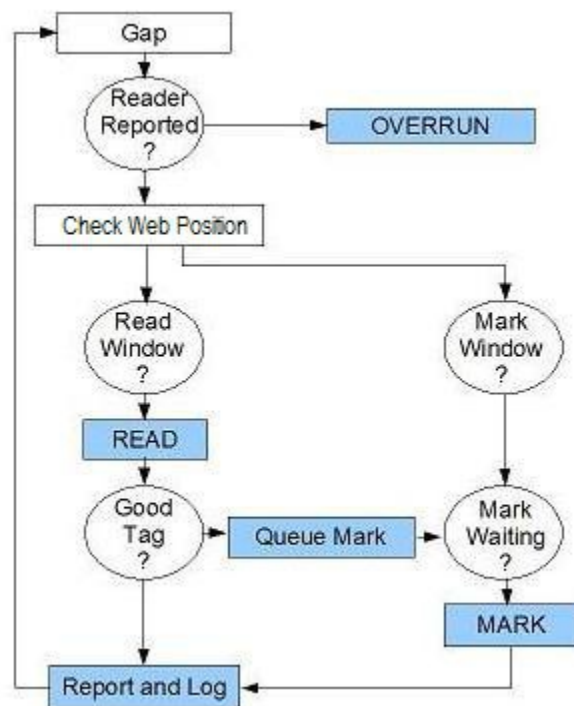
This is a high level flowchart of a typical LineLogix application.

#### 9.1.1. Checking Web Position

The Propeller chip in the TCU has a core dedicated to monitoring the Gap input in a tight look at 80 MHZ.

When the Gap sensor changes state to its active condition as set by the InvertGap flag, multiple actions trigger nearly immediately:

- If the reader has not reported from the last cycle, an Overrun error can be posted to the PC and the previous tag marked bad.
- If a position encoder is equipped (rare), further actions may be delayed until the label is in a configured position.
- In most applications, the Check Web Position and Window blocks of the diagram falls through immediately and further actions are effects of the Gap input.
- The test command (in the diagram a READ) is sent to the connected RFID reader.
- In the same cycle, if the Marking subsystem indicates that a bad label is under the marker, the Mark output is set.



#### 9.1.2. Window Determination

In Encoder systems (rare), the secondary activity of the TCU is to constantly compare the Web Position to programmed setpoints (see the message protocol below) and trigger activities as configured. Setpoints are provided for the start and end of the read and mark

“windows,” where the “window” is the web position at which the RFID reader and marking device are triggered. Setpoints are expressed in Encoder counts. Each setpoint is the distance traveled by the web between the end of the last gap and the event determined by the setpoint. If the gap sensor were always positioned at the upstream edge of the test section, the windows would simply be the distance between the gap and the desired action location, expressed in Shaft Encoder counts.

**In systems with no Encoder, all setpoints are 0, i.e., all actions are triggered by the leading edge of a label in the Gap Sensor.**

### 9.1.3. Marking

When a bad tag is detected, it is not necessarily marked while still in the test section or even the RIP. It may be marked up to 64 label locations downstream of the test section, based on the configuration parameter **Sense Mark Delta**.

### 9.1.4. Overruns

If, at the time any Read Window opens, the previous label's test result has not been reported by the RFID reader, an Overrun condition exists. This condition may indicate damage to RFID reader, a loss of power/configuration to reader or TCU, or an overspeed condition. If the TCU is configured to show Overrun errors, the red light will be activated; if so configured, the web will also be stopped.

When an Overrun has been detected, the Overrun Detect bit will be set in the Query reply until it is cleared by a Stop Off (s00!<CRLF>) command.

Overrun detection may be suppressed by a Flag bit, see the message protocol below. This may be important to prevent false reports in high speed, close pitch applications.

## 9.2. TCU I/O Map

### 9.2.1. SX48 TCU 2.x

The SX48 I/O pins are mostly unallocated on the standard TCU, to allow the greatest possibility for custom applications. The allocated I/O pins are as follows:

<i>Port</i>	<i>Pin</i>	<i>LED</i>	<i>Function</i>
A	0	none	Serial TX to PC through level converter
A	1	none	Serial RX from PC through level converter
B	1	Yellow	Test Trigger Output to RFID Reader
C	1	Green	Good Test Result Input from Reader
C	2	Red	Bad Test Result Input from Reader
D	0	Red	Gap Input from Sensor through Grayhill module
D	1	Red	Stop Output through Grayhill module

<i>Port</i>	<i>Pin</i>	<i>LED</i>	<i>Function</i>
D	2	Red	Mark Output through Grayhill module
D	3	Red	Overrun Output through Grayhill module
D	4	Red	Bad Tag Output through Grayhill module
D	5	Red	Good Tag Output through Grayhill module
D	6	Red	Shaft (Label Position) Encoder Phase A Input OR Serial TX to Reader at TTL Level
D	7	Red	Shaft (Label Position) Encoder Phase B Input OR Serial RX from Reader at TTL Level

### 9.2.2. Propeller, TCU 3.x

Propeller I/O is organized through three 32-bit arrays, DIRA, OUTA and INA. Parallax seems to plan a second I/O bank at DIRB, etc., for future expansion. TCU 3.x firmware defines its I/O as follows:

```

pLLIOHostTx           = 0
pLLIOHostRx           = 1
pLLIOSerialPassthrough = 2
pLLIOGapIn            = 3
pLLIOStopOut          = 4
pLLIOMarkOut          = 5
pLLIOOverrunOut       = 6
pLLIOBadOut           = 7
pLLIOGoodOut          = 10
pLLIOTargetRx         = 17
pLLIOTargetTx         = 18
pLLIOBarcodeRx        = 19
pLLIOBarcodeTx        = 21

```

### 9.2.3. Propeller, TCU 4.x

```

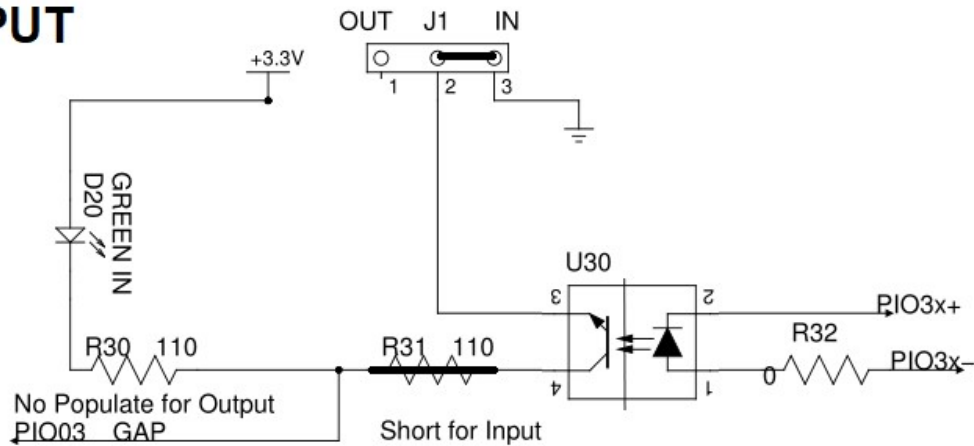
pLLIOHostTx           = 0
pLLIOHostRx           = 1
pLLIOSerialPassthrough = 2
pLLIOGapIn            = 3
pLLIOStopOut          = 4
pLLIOMarkOut          = 5
pLLIORedOut           = 6
pLLIOBadOut           = 7
pLLIOGoodOut          = 10
pLLIOGapEcho          = 14
pLLIOBarcodeRx        = 16
pLLIOBarcodeTx        = 20
pLLIOVarPrnRx         = 19
pLLIOVarPrnTx         = 21
pLLIOTargetRx         = 17
pLLIOTargetTx         = 18
pLLIOHFTTargetRx      = 22
pLLIOHFTTargetTx      = 29
pLLIOPaperOutInput    = 27
pLLIOInlayBadInput    = 26
pLLIODebugUsbTx       = 30
pLLIODebugUsbRx       = 31

```

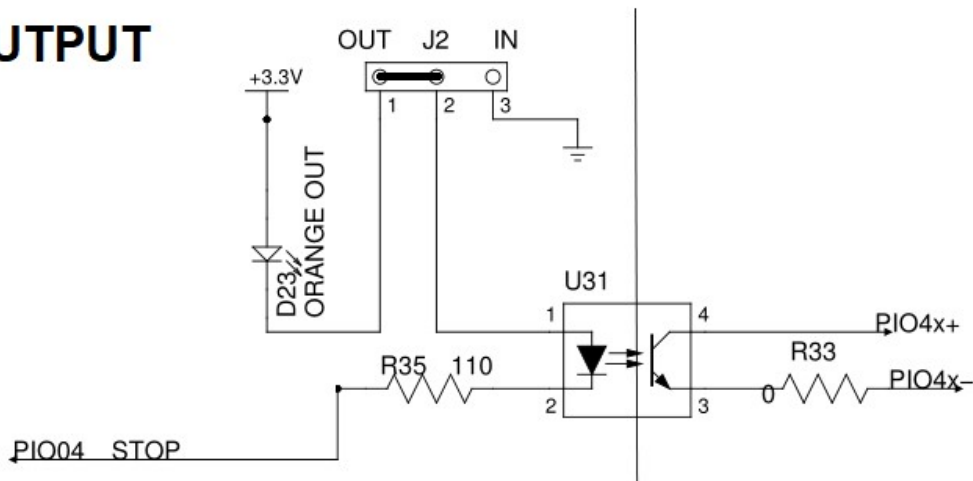
### 9.2.4. TCUV4 Equivalent I/O Circuits

TCUV4 uses LTV816 4-pin optoisolator chips for I/O. The chips are socketed for easy repair. Input circuits use the LTV816 “upside down” with pin 1 at the bottom right, so the +- polarity of inputs is different from that of outputs. Standard single PNP/NPN inputs must be current limited with an external resistor in order to avoid burning out the LTV816. Multiple LTV816 inputs can be chained, and the resistor reduced or discarded.

#### INPUT



#### OUTPUT



### 9.3. TCU Serial Protocol

The TCU communicates over a serial port with no handshaking but with echo. The first generation TCU (1.x) for SAMSys readers had a serial port separate from the reader's, running at 9600 baud. The 2.x and 3.x TCU, for ThingMagic and Feig readers, shares serial port with the reader. PCs connect to the modern TCU by releasing DTR, and connect through to the reader by asserting DTR. LineLogix systems run at 38400 or 115200 baud, always 8N1: 8 data bits, No parity, 1 Stop bit.

### 9.4. TCU Commands

Each TCU command consists of 4 human readable characters followed by a CRLF line termination:

<i>Byte 0</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>Termination</i>
Opcode	Parameter High Nibble	Parameter Low Nibble	'!' or Checksum	CRLF

<i>Opcode</i>	<i>Parameter</i>	<i>Functionality</i>
0 . . F	Value to write to memory location	Parameter command for TCU. The Parameter is written to the memory location mapped by the opcode.
K	Ignored	TCU 2.x/3.x Only: Calculate CRC of target command
L	Light Pattern	Test Lights
M	Ignored	Trigger a Mark operation
m	ignored	Remove the mark trigger and restore normal operation
O	ignored	Query TCU I/O state (Reply details below)
P	ignored	Pause the automatic operation of the TCU by setting bit 0 of the Flags byte
p	ignored	Resume (unpause) automatic operation.
Q	2.x: ignored 3.x: parameters	Query TCU memory state (Reply and Parameter details below)
R	ignored	Trigger a Read (Test) operation
r	ignored	Remove Read trigger and restore normal operation
S	ignored	Stop the web
s	ignored	Restart the web
V	ignored	Get TCU Software Version
Z	Varies	Used to configure the Stop On Count target value. Refer to the Stop On Count section.

<i><b>Opcode</b></i>	<i><b>Parameter</b></i>	<i><b>Functionality</b></i>
>	Inlay Status	Used to report good or bad tag status to the TCU in PC controlled applications. 00 indicates a bad inlay while 01 indicates a good inlay.

#### 9.4.1. Stop On Count

The TCU can be configured to halt operation after it has processed a certain number of inlays. Configuration is done with the Z TCU Command. The parameter of Z TCU command has several different meanings:

<i><b>Parameter</b></i>	
0x00	Resets the current tag count and turns off Stop On Count.
0x01 - 0x04	Use the value of the next command to set a byte of the Stop On Count target value. Issuing the following commands will set the Stop On Count target value to 0xAABBCCDD: Z01! ZDD! Z02! ZCC! Z03! ZBB! Z04! ZAA!
0xFF	Return the current tag count.

#### 9.4.2. TCU Parameter Memory

<i><b>Location</b></i>	<i><b>Parameter</b></i>	<i><b>Functionality</b></i>
0	Shaft Terminal Count Low	Loaded into current Shaft Encoder register when a gap is sensed with the web going backwards. Currently unused. TCU 2.x: Unused TCU 3.x: Barcode Delta
1	Shaft Terminal Count High	
2	Read Window Start Low	Shaft Encoder position at which the Read (Test) Trigger is set TCU 2.x/3.x: Write Increment Addend (2), Start Byte in Target Command for Increment Operation (3)
3	Read Window Start High	

<i><b>Location</b></i>	<i><b>Parameter</b></i>	<i><b>Functionality</b></i>
4	Read Window End Low	Shaft encoder position at which the Read (Test) Trigger is cleared
5	Read Window End High	TCU 2.x/3.x: Number of Bytes to carry Increment operation (4), Gap Skip Count (5), Gap Skip Count Calibration Enable (byte 5, bit 7)
6	Mark Window Start Low	Shaft encoder position at which the Mark trigger is set
7	Mark Window Start High	TCU 2.x/3.x: Number of milliseconds to wait for reply before stopping (6, default 0 or Never), Unused (7)
8	Mark Window End Low	Shaft encoder position at which the Mark Trigger is cleared. If the Mark By Time feature (Parameter byte B bit 7) is enabled, this value is used to load a 16-bit counter at the time Mark is triggered; the counter decrements about every 4 uS, and the Mark signal is removed when it counts down to zero. TCU 2.x/3.x: Always used for Mark By Time
9	Mark Window End High	
A	Sense Mark Delta	Number of labels between Test Antenna and Marking Device
B	Flag Byte 1	See Below
C	Flags Byte 2	See Below
D	Flags Byte 3	See Below
E	Flags Byte 4	See Below
F	Flags Byte 5	See Below

#### **Flags Byte 1 (Parameter 0x0B)**

<i><b>7</b></i>	<i><b>6</b></i>	<i><b>5</b></i>	<i><b>4</b></i>	<i><b>3</b></i>	<i><b>2</b></i>	<i><b>1</b></i>	<i><b>0</b></i>
Mark By Time	Mark Immediate	Stop On Overrun	Use Shaft Encoder	Stop On Mark	Stop On Read	Check Overruns	Disable Auto

#### **Flags Byte 2 (Parameter 0x0C)**

<i><b>7</b></i>	<i><b>6</b></i>	<i><b>5</b></i>	<i><b>4</b></i>	<i><b>3</b></i>	<i><b>2</b></i>	<i><b>1</b></i>	<i><b>0</b></i>
Windows Span Gap	Notify On Read	RFU	Invert Gap	Invert Overrun	Invert Stop	Invert Mark	Invert Read

**Flags Byte 3 (Parameter 0x0D)**

<i>7</i>	<i>6</i>	<i>5</i>	<i>4</i>	<i>3</i>	<i>2</i>	<i>1</i>	<i>0</i>
Lights Off on PLC Unread	Stop Count on Good Read	Suppress Target Command	Count Job	Stepper Attached	RFU	Write Target Command Bytes 8..15	Write Target Command Bytes 0..7

**Flags Byte 4 (Parameter 0x0E)**

<i>7</i>	<i>6</i>	<i>5</i>	<i>4</i>	<i>3</i>	<i>2</i>	<i>1</i>	<i>0</i>
Use HF Reader	Write To Compare Buffer Bytes	Adjust Thing Magic Timeout	Verify From Bar-code	Write From Barcode	Bardcode is Decimal	Write To Verify Buffer Bytes 8..15	Write To Verify Buffer Bytes 0..7

**Flags Byte 5 (Parameter 0x0F)**

<i>7</i>	<i>6</i>	<i>5</i>	<i>4</i>	<i>3</i>	<i>2</i>	<i>1</i>	<i>0</i>
RFU	RFU	RFU	RFU	RFU	Suppress Reader Echo	Use Operation Table	Ignore Fast Gap

Notes:

**Bit 3, Stepper Attached:** Pulses Stop output when False per the InvertStop bit, creating a Go signal for a stepper controller.

**Bit 4, Count Job:** Continually sends Read commands, counts hits in window set by Gap input, reports when Gap goes False per the InvertGap bit. PLC systems can define a read window for this function by raising the Gap signal when the inlay is near the antenna and releasing the Gap signal when other inlays might interfere with singulation.

**Bit 5, Suppress Target Command:** Used with >Raction in the LineLogixPC jobfile to implement multistep operations (like Write, the Lock) on transports capable of stopping for each tag.

**Bit 6, StopOnGood:** On Count job, stop the repeated reads as soon as a good tag is detected. The Count is therefore always 0 or 1. Makes the Count job more like a Read job, but still allows multiple tries on a long read window.

**Bit 7, LightsOffUnread:** Turn off lights and PLC good/bad feedback signals when the read window closes. Used when the PLC needs the Good signal to be secured before the read window is opened, to avoid false good status at the start of the read window.



### 9.4.3. TCU Replies

Commands with unknown opcode draw a NAK: ?<CRLF>

Parameter commands draw an ACK: !<CRLF>

Most IO commands (M, m, O, R, r, S and s) draw the IO reply:

Obbcc<CRLF>

where bb is the contents of IO Port B and cc is Port C

The Query reply is structured as:

gxx<CRLF>	TCU Gap Counter Low Byte
Gxx<CRLF>	TCU Gap Counter High Byte
lxx<CRLF>	Last Label Length in Counts, Low Byte
Lxx<CRLF>	Last Label Length in Counts, High Byte
sxx<CRLF>	TCU Shaft Encoder Position Low Byte
Sxx<CRLF>	TCU Shaft Encoder Position High Byte
Mxx<CRLF>	TCU Operation Mask (flags)

Where xx is the hex value associated with each parameter.

The bits of the Mask byte (Mxx) are:

7	6	5	4	3	2	1	0
GoodTag	BadTag	TCU 2.x/3.x: Target Cmd Sent	TCU 2.x/3.x: Target Replied	Overflow	Backing	RFU	RFU

Commands with improper framing or checksum are ignored.

### 9.4.4. Notifications

Asynchronous notifications may be sent over the serial line by the TCU. Such Notifications start with a Greater Than sign (“>”) and end with a line termination. Between those two framing elements, the Notification will begin with a one-letter code and may include other information.

#### Overflow

The host PC is always notified of Overruns via the serial port:

>O<CRLF>

## Read

If Notify on Read (bit 6 of Flags byte 0x0C) is set, the Read Notification will be sent whenever a serial command is sent to the reader:

>Rgggg<CRLF>

Where gggg is the current Gap count in hex.

## Count

If Count (bit 4 of Flags byte 0x0D) is set, the Read Notification will be sent whenever a serial command is sent to the reader:

>Cccgggg<CRLF>

Where gggg is the current Gap count in hex, and cc is the count of successful reads of the current tag in hex.

## Operation Table Response

If Use Operation Table (bit 1 of Flags byte 0x0F) is set, the Operation Table Response will be sent after the actions listed in the Operation Table have been executed.:

>Tssgggguuuuuuuuuuuuuuuuuuuu, <DATA><CRLF>

Where ss is the inlay status (00 for bad, 01 for good), gggg is the current Gap count in hex, uuuuuuuuuuuuuuuuuuu is the inlay's unique identifier, and <DATA> is an optional field containing any additional data associated with this inlay.

### 9.4.5. Example

A typical TCU job, which might appear in a LineLogixPC job file, is:

```
220!  
300!  
428!  
500!  
628!  
700!  
800!  
908!  
BB0!
```

Which sets the read window to between 0x20 and 0x28 counts (2.14 inches through 2.67 inches), and the mark window to 0x0800 4-uS counts (around 8 mS) starting at 0x28 counts (2.67 inches). The Flag bits are **Mark By Time**, **Stop On Overrun** and **Use Shaft Encoder**. Since this job does not specify a Sense Mark Delta, the marking device would have to be positioned over the test section.

## 9.5. S Records

TCU memory can alternately be read and written by a mechanism called S-Records in development, though the record structure is closer to Intel Hex records:

To read memory the record starts with a semicolon:

;0000000010

Reads 0x10 (=16) bytes from the start of memory. That is the entire Parameter memory detailed above.

To write memory, the record starts with a full colon:

:000000002082000280028000008B0

Duplicates the commands 220! through 908! above.

The introduction of S records in 2011 greatly speeded TCU setup, especially in systems with multiple TCUs.

## 9.6. Operation Table

The operation table allows the TCU to perform complex actions on each inlay. For instance, the TCU can be configured to inventory an inlay, read a block of user memory, and verify that the data is correct.

Each entry in the Operation Table looks like:

<i>Byte 4</i>	<i>3</i>	<i>2</i>	<i>1</i>
RFU	Length	Address	Operation Code

### 9.6.1. Operation Codes

Operation	Value	Address	Length
Thing Magic Inventory	0x00	Ignored	Ignored
Thing Magic Read Single ID	0x01	Offset of TID in Reader's Last Response	Length of TID
ThingMagic Write EPC from Barcode	0x02	If nonzero, sets start of EPC data in RFID RX buffer	If nonzero, sets length of data to copy from RFID RX to TX buffers.
Feig Inventory	0x03	Ignored	Ignored
Feig Select	0x04	Ignored	Ignored
Feig Read	0x05	Data block to Start Read	Number of Bytes to Read
Emit To Barcode	0x06	Offset of Reader's Last Response	Number of Bytes of Reader's last response to emit to barcode.

Thing Magic Clear Tag Buffer	0x07	Ignored	Ignored
Thing Magic Read Multiple TID	0x08	Ignored	Ignored
Thing Magic Get Tag Buffer Entry	0x09	Ignored	Ignored
Thing Magic Read TID Single	0x0A	Ignored	Ignored
Save Response to Current Barcode Buffer	0x0B	Offset of Reader's Last Response	Number of Bytes of Reader's last response to save to barcode buffer.
Reserved	0x0C		
Reserved	0x0D		
Reserved	0x0E		
Reserved	0x0F		
Reserved	0x10		
Reserved	0x11		
Reserved	0x12		
Reserved	0x13		
End Sequence (Good Tag)	0x14	Ignored	Ignored
Feig Write Data	0x15	Data block to Start Write	Number of Bytes of Target Compare Buffer to Write to Inlay
Compare	0x16	Offset of Reader's Last Response	Number of Bytes of Target Compare Buffer to Compare to Reader's Last Response

## 10. Glossary and Acronym List

0x – Prefix of a hexadecimal (base 16) number

CRLF, <CRLF> – ASCII Carriage Return (10) followed by ASCII Line Feed (13)

FPM – Feet Per Minute

I/O – Input and/or Output

mS – Milliseconds

RIP – RFID Instrument Package

RFU – Reserved for Future Use

TCU – Timing Control Unit

Test Section – Region of space where RFID tests are conducted. This is the region in the plane of the RFID web, between the RF Baffles of the antenna enclosure.

uS – Microseconds