



Greetings ...

This file contains the answers to our **Email PLC Quiz #215**. This edition is a **Beyond Beginner** quiz which focuses on the differences between the "Toggle Bit" and "Force" features of RSLogix software. It also covers some of the differences between the Allen-Bradley PLC-5, SLC-500, and ControlLogix platforms which are related to the use of these features.

If you'd like to discuss the information contained in any of our quizzes, please feel free to contact us. We'll be glad to answer any questions that you might have.

The following equipment was used during the research and development of this edition of the Email PLC Quiz. Other configurations might possibly give different results.

PLC-5/20E Processor; Series E; Revision D.2
RSLogix5 Software; Version 6.00.00

SLC-5/04 Processor; Series C; OS401
RSLogix500 Software; Version 6.00.00

Note: Most MicroLogix1000 Processors will duplicate the SLC's operation

ControlLogix5555 Processor; Revision 15.4
RSLogix5000 Software; Version 15.00.00

Communication through DF1 Serial Port Connection; 19,200 Baud;
RSLinx Lite Software; Version 2.50.00.20 (CPR 7)

Microsoft Windows 2000 Software; Version 5.00.2195

A number of readers have responded to previous editions of our Email PLC Quizzes with requests for any books that we might have available covering this type of material. At this point in time, we are providing training mainly through our five-day PLC Boot Camp classes. So far we haven't found a satisfactory way to effectively present the same amount of detail in any format other than through face-to-face hands-on training. So no, we're sorry that we don't have any books or other types of training materials to offer - but we thank you very much for asking.

Beyond Beginner Quiz #215 - Answers to the Questions

Answer 1A - for a PLC-5 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

This was intended to be a very simple "warm up" question for the PLC-5 category. If you missed it, go back to this month's **Beginner Level Quiz #115** for a review. Basic ideas: The toggle operation at Position B will write a ONE into the Pump's bit on the Output Data Table. The common "seal in" arrangement of rung #0000 will then keep that ONE in place. Each time the processor scans rung #0001, the processor will evaluate the XIC at Position D as TRUE - and then execute the OTE at Position F with TRUE logic. This will write a ONE into the Horn's bit. Pressing the Stop_Button will "unseal" rung #0000 - and result in turning OFF both the Pump and the Horn.

Answer 2A - for an SLC-500 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

This was intended to be a simple "warm up" question for the SLC-500 category. This SLC-500 question has EXACTLY the same answer - for EXACTLY the same reasons - as Question 1 for the PLC-5 platform. If you missed it, go back to this month's **Beginner Level Quiz #115** for a review.

Answer 3A - for a ControlLogix system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

This question for the ControlLogix platform is slightly more complicated than Questions 1 and 2 above. The results will PROBABLY be the same - but there is a distinct possibility that rung #0000 will not be "sealed in" when the Pump's bit is toggled. The reason is that the ControlLogix system will allow a "Toggle Bit" operation to reach a memory location at ANY time - including PART WAY through the execution of a rung. (PLC and SLC systems normally don't allow this.) Suppose that the "Toggle Bit" signal happens to reach the Pump's output bit JUST AFTER the processor has executed the XIC at Position C. In that case, the XIC will have already been evaluated as FALSE. Then the processor will execute the OTE at Position B with FALSE logic - and the OTE will write a ZERO into the Pump's bit. In other words, our "Toggle Bit" operation just got overwritten - and the Pump will stay OFF - and the Horn will NOT sound.

So what are the odds? When I experimented with these conditions in the lab, about one or two random tries out of a hundred would fail to bring on the Pump and the Horn. So if you're going to bet on the outcome of this one, it's probably OK to bet your lunch money - but I wouldn't bet the rent.

Answer 4A – for a PLC-5 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

Notice that the bit we're toggling this time has an INPUT (I:___) address. First things first. Can we – or can't we – toggle a bit with an INPUT address? Yes, we can. Specifically, we CAN send a signal to the PLC-5 processor and tell it to toggle (in other words, to reverse) the status of an INPUT bit. The tricky part is that the bit won't stay in this toggled state for very long. That's because just before the processor starts every scan of the ladder logic rungs, it UPDATES the Input Data Table. The Start_Button in the field is OFF – and so just before the processor starts each pass through the rungs, it writes a ZERO into the Start_Button's bit.

Then somewhere along the line, we come along and manually toggle that bit to a ONE status. (With a PLC-5, this can happen at any point during the processor's pass through the ladders – but not partway through a rung.) Then – before the very next pass through the ladder rungs – the processor puts a ZERO right back into the bit. So the question now becomes: What status (ZERO or ONE) will be stored in the bit at the instant that the processor executes the XIC at Position H? Place your bets – and spin the wheel. It's actually a random thing. But – in a way – I've rigged the wheel. With a very short program like the one that we're using in this quiz, the odds are VERY good that the XIC at Position H will PROBABLY see the ONE status that we've manually toggled into the Start_Button's bit. If that happens, then the Pump will be "sealed" ON – and the Horn will sound. The key word is "probably" because we're dealing with an INPUT bit in this question. Still, with the short program being used in our quiz, the odds are VERY good that the Horn will sound continuously – at least for this question concerning the PLC-5 platform.

Now suppose that the program is much longer than the one that we're using. Things get a lot trickier. If the "seal in" rung for the Pump happens to be located near the END of a long program, then the toggle operation will PROBABLY turn the Pump ON – and the Horn will sound. On the other hand, if the "seal in" rung for the Pump happens to be located near the BEGINNING of a long program, then the toggle operation PROBABLY won't turn the Pump ON – and the Horn will NOT sound. We can't go into all of the details here, but ask yourself how big a "target" we'll have to shoot for – between our toggle operation – and the processor's normal update of the data bit. (Longer program = bigger target.) Then ask which position within that target would give the processor a better chance to execute the XIC in its toggled state. (Near the end = better chance.)

The "Toggle Bit" operation is a very handy feature, but by this point you should have a good idea of why many technicians really don't understand it well enough to use it effectively while troubleshooting a system. The thing seems so RANDOM that learning how it works through trial-and-error experience is quite often a losing proposition. The best course of action is usually to avoid toggling any bits with an INPUT (I:___) type address. Toggling other types of bits invariably gives much more reliable results. Incidentally, our PLC Boot Camp classes cover other techniques which overcome the random nature of toggling input bits.

As we said before, this discussion isn't intended to be an actual "lesson" – but hopefully we've covered enough about the concepts involved to explain the answer to this particular question in our quiz.

Answer 5C – for an SLC-500 system - The Horn probably will NOT sound continuously. Later if the Start_Button is pressed, the Horn will turn ON.

Notice that this answer for the SLC-500 platform is different from the PLC-5 system that we just discussed. Again - first things first. Can we - or can't we - toggle a bit with an INPUT address? Yes, we can. Specifically, we CAN send a signal to the SLC-500 processor and tell it to toggle (in other words, to reverse) the status of an INPUT bit. The tricky part is that the bit won't stay in this toggled state for very long. That's because just before the processor starts every scan of the ladder logic rungs, it UPDATES the Input Data Table. The Start_Button in the field is OFF - and so just before the processor starts each pass through the rungs, it writes a ZERO into the Start_Button's bit.

So far the SLC-500's operation is EXACTLY the same as we covered for the PLC-5 in our previous question. Now for the difference. Unlike the PLC-5, the SLC-500 will NOT let us toggle a bit at any point during the processor's pass through the rungs. Specifically, we can only toggle the bit AFTER the processor executes all of the rungs - but BEFORE the next automatic update of the bits on the Input Data Table.

So regardless of the fact that we CAN toggle the bit for the Pump's Start_Button to a ONE condition, the XIC at Position H will NEVER see that toggled status. Instead, it always sees the actual ZERO status of the Start_Button in the field - and so the XIC gets executed as FALSE. The Pump's output bit stays in its ZERO status - and the Horn will NOT sound.

Once again, the "Toggle Bit" operation is a very handy feature, but here is another reason why many technicians don't understand it. In some cases, using the same operation on the SLC-500 platform may give totally different results from using the identical operation on a PLC-5 system.

As we said before, probably the best course of action is just to avoid toggling any bits with an INPUT (I:__) type address. Toggling other types of bits invariably gives much more reliable results. And once again, this discussion isn't intended to be an actual "lesson" - but hopefully we've covered enough about the concepts involved to explain the answer to this particular question in our quiz.

Answer 6A – for a ControlLogix system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

This ControlLogix question has the same answer as for the PLC-5 platform covered in Question #1 - but for a different reason. Unlike the PLC-5 and SLC-500 platforms, the ControlLogix doesn't update its Input Data Table just before each pass through the ladder logic program. Instead the input data is updated in the processor at a regularly scheduled period called the RPI (Requested Packet Interval) which is set up in the configuration of the input module.

The important thing to consider is that the periodic update of the input data can take place at ANY point during the scan of the ladder logic – including even PART WAY through the execution of a rung. This introduces even more random chances for our toggle action at Position J to “hit” or to “miss” affecting our test program.

Let’s work through this. The Start_Button in the field is OFF – and so each time the RPI (update period) elapses, a ZERO is written into the Start_Button’s bit. Then somewhere along the line, we come along and manually toggle that bit to a ONE status. On the very next regularly scheduled update, the Start_Button in the field is still OFF – and so a ZERO gets written right back into the Start_Button’s bit again.

So the question now becomes: What status (ZERO or ONE) will be stored in the bit at the instant that the processor executes the XIC at Position H? Place your bets – and spin the wheel. It’s actually a random thing – but the wheel is NOT always fair.

Consider how a short program (as in our quiz) as opposed to a long program (as in an actual control system) might affect our “Toggle Bit” operation. As the ControlLogix processor loops around and around through a short program, we’d have MORE chances for the XIC to execute while our bit stays in its toggled state – and “seal in” our Pump control rung. So with the short program of our quiz conditions, the Horn PROBABLY would sound. On the other hand, as the processor passes through a long program, we’d have LESS chances for the XIC to execute before our bit gets “refreshed” to its non-toggled state – and miss “sealing in” our Pump control rung. So with a long program, the Horn might NOT sound.

Now consider how a longer – or a shorter – RPI (update rate) might affect our “Toggle Bit” operation. With a longer update period, we’d have MORE chances for the XIC to execute with our bit in its toggled state. While experimenting for our quiz, I left the RPI at its default setting of 10 milliseconds – and the Horn sounded about 98% of the times that I ran the test. When I set the RPI for only 0.2 milliseconds, the Horn sounded less than 1% of the times that I ran the same test. That’s because at the shorter (more frequent) update rate, the bit was much more likely to reflect the actual ZERO status of the Start_Button device in the field.

Now that we’ve discussed its importance, you can see why the questions for our quiz specified the default setting of 10.0 for the RPI.

And so here we have another reason why many technicians don’t understand – or trust – the “Toggle Bit” feature well enough to use it for many of their troubleshooting problems. That’s unfortunate because it’s really a handy tool. As long as we stay away from toggling INPUT bits, then things pretty well make sense. So for many people, the best course of action is usually to avoid toggling any bits with an INPUT type address. Toggling other types of bits invariably gives much more reliable results. And once again, we do cover other techniques which overcome the random nature of toggling input bits in our PLC Boot Camp classes.

While this discussion isn’t intended to be an actual lesson, hopefully we’ve covered enough about the concepts involved to explain the answer to this particular question in our quiz.

Answer 7A – for a PLC-5 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

This question is EXACTLY the same as Question 4. Many technicians mistakenly think that a "Toggle Bit" operation performed on the Input Data Table is somehow different from a "Toggle Bit" done on an instruction (for example: on an XIC) which has the same address. There is NO difference. Remember that we're ALWAYS toggling a BIT – in other words, a BOX – located in the processor's memory – and NEVER an instruction located in the ladder logic rungs.

Answer 8C – for an SLC-500 system - The Horn probably will NOT sound continuously. Later if the Start_Button is pressed, the Horn will turn ON.

This question is EXACTLY the same as Question 5. Many technicians mistakenly think that a "Toggle Bit" operation performed on the Input Data Table is somehow different from a "Toggle Bit" done on an instruction (for example: on an XIC) which has the same address. There is NO difference. Remember that we're ALWAYS toggling a BIT – in other words, a BOX – located in the processor's memory – and NEVER an instruction located in the ladder logic rungs.

Answer 9A – for a ControlLogix system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

This question is EXACTLY the same as Question 6. Many technicians mistakenly think that a "Toggle Bit" operation performed on the Input Data Table is somehow different from a "Toggle Bit" done on an instruction (for example: on an XIC) which has the same address. There is NO difference. Remember that we're ALWAYS toggling a BIT – in other words, a BOX – located in the processor's memory – and NEVER an instruction located in the ladder logic rungs.

Answer 10A – for a PLC-5 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

First things first. Here we're toggling a bit with an output (O:__) type address. After the toggle operation, the original ZERO status of the Pump's bit will be reversed to a status of ONE. The "seal in" construction of rung #0000 will then maintain that ONE status of the Pump's bit. Then the XIC at Position D will test TRUE – and so the OTE at Position F will write a ONE into the Horn's bit. And the Horn will sound continuously.

So what about those forces on the Pump? Secret handshake: They have absolutely NO EFFECT on the results of this question. As discussed later, forces are applied to OUTPUT signals DOWNSTREAM of the output bits. Specifically, a force applied to an OUTPUT will NOT affect the operation of the ladder logic program. Now the device in the field (for example: our Pump) WILL be affected by the force – but NOT the status of the output bit in the processor's data table.

Over the years our PLC Boot Camp courses have helped MANY students understand this concept. Usually the technicians with the most previous experience are the hardest to convince. They often come into our classes with the attitude that they already understand "quite a lot" about PLCs – including how a simple force works. Finding out that there is more to the subject can sometimes be a "trying" revelation. Regardless of what they've believed for years, a series of hands-on experiments like the one being used for this question of our quiz can always prove the point.

Answer 11A – for an SLC-500 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

In this particular test, the SLC-500 platform will function EXACTLY the same as the PLC-5 system which we discussed for Answer 10A – and for EXACTLY the same reasons.

Answer 12A – for a ControlLogix system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

In this test, the ControlLogix will PROBABLY work just like the other systems in the last two questions. But there could be a catch. Remember that the ControlLogix system will allow a "Toggle Bit" operation to reach a memory location at ANY time – including PART WAY through the execution of a rung. (PLC and SLC systems normally don't allow this.) Suppose that the "Toggle Bit" signal happens to reach the Pump's output bit JUST AFTER the processor has executed the XIC at Position C. In that case, the XIC will have already been evaluated as FALSE. Then the processor will execute the OTE at Position B with FALSE logic – and the OTE will write a ZERO into the Pump's bit. In other words, our "Toggle Bit" operation just got overwritten – and the Pump will stay OFF – and the Horn will NOT sound.

If this sounds familiar, it's because we covered EXACTLY the same concepts in Question 3. The main thing to realize here is that the forces on the Pump's output have absolutely NO effect on how the system responds. That detail surprises many technicians – even some "Top Guns" with years of experience.

Answer 13A – for a PLC-5 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

The answer for Question 13 is EXACTLY the same as the answer for Question 10 – for EXACTLY the same reasons. Specifically, toggling at Position D acts upon the Pump's bit/box in the processor's memory – EXACTLY the same as toggling at Position B. Remember that we're NOT toggling an INSTRUCTION (example: OTE, XIC, etc.) – instead we're toggling a BIT in the processor's memory.

Answer 14A – for an SLC-500 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

The answer for Question 14 is EXACTLY the same as the answer for Question 11 – for EXACTLY the same reasons. Specifically, toggling at Position D acts upon the Pump's bit/box in the processor's memory – EXACTLY the same as toggling at Position B. Remember that we're NOT toggling an INSTRUCTION (example: OTE, XIC, etc.) – instead we're toggling a BIT in the processor's memory.

Answer 15A – for a ControlLogix system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

The answer for Question 15 is EXACTLY the same as the answer for Question 12 – for EXACTLY the same reasons. Specifically, toggling at Position D acts upon the Pump's bit/box in the processor's memory – EXACTLY the same as toggling at Position B. Remember that we're NOT toggling an INSTRUCTION (example: OTE, XIC, etc.) – instead we're toggling a BIT in the processor's memory.

Answer 16A – for a PLC-5 system - The Horn probably WILL sound continuously. Later if the Stop_Button is pressed, the Horn will turn OFF.

So which is more powerful – a "Toggle Bit" operation – or a "Force" operation? This exercise shows that (at least for the PLC-5 platform), toggling will override a force in most cases. Since we're toggling a bit with an INPUT address, the same issues that we covered in Answer 4A (program scan time, the rung's location in the program, etc.) still apply. But the main idea here is that even though the Start_Button is being forced OFF, we can still toggle the bit to a ONE condition – at least until the next update of the Input Data Table. And if the processor executes the XIC at Position H while our ONE is still in the bit/box, then the Pump's rung will "seal in" and the Horn will sound.

Many students (especially the most experienced ones) find this to be a totally "mind blowing" demonstration. For years they've been told that "You can't toggle an INPUT bit" - and yet that "rule" is obviously not true. They've always believed that a "Force" is a powerful operation that overrides any other condition - and yet here simply toggling a bit completely defeats the force operations on TWO field devices (both the Start_Button and the Pump.)

In our PLC Boot Camp classes, it's extremely common for students with several years of experience to tell us that demonstrations such as this one have cleared up issues which had been confusing them for years.

Answer 17D - for an SLC-500 system - The Horn probably will NOT sound continuously. Later if the Start_Button is pressed, the Horn will stay OFF.

Notice that this answer for the SLC-500 platform is different from the PLC-5 system that we just discussed.

At first glance, it appears that forcing the Start_Button OFF prevents us from being able to toggle the input bit. But that's not exactly what's happening here. As we covered in Answer 8C, we ARE able to toggle the bit and reverse its status to a ONE - but then just before the processor starts the next pass through the ladder logic rungs, it UPDATES the Input Data Table. Since the Start_Button signal is being forced OFF, the processor starts each pass through the rungs with a ZERO in the Start_Button's bit.

Just to nail this down, remember that unlike the PLC-5, the SLC-500 will NOT let us toggle a bit at any point during the processor's pass through the rungs. Specifically, we can only toggle the bit AFTER the processor executes all of the rungs - but BEFORE the next automatic update of the bits on the Input Data Table.

So regardless of the fact that we CAN toggle the bit for the Pump's Start_Button to a ONE condition, the XIC at Position H will NEVER see that toggled status. Instead, it always sees the forced ZERO status of the Start_Button signal - and so the XIC gets executed as FALSE. The Pump's output bit stays in its ZERO status - and the Horn will NOT sound.

Here's another case where using an operation on the SLC-500 platform gives totally different results from using the identical operation on a PLC-5 system. This is just one more reason why many technicians don't understand - or use the handy "Toggle Bit" feature.

Answer 18D - for a ControlLogix system - The Horn probably will NOT sound continuously. Later if the Start_Button is pressed, the Horn will stay OFF.

Notice that this answer for the ControlLogix platform is the same as the SLC-500 system that we just discussed - but for a different reason.

In a ControlLogix system, forcing the Start_Button OFF actually DOES prevent us from being able to toggle the input bit. Regardless of the fact that our "Toggle Bit" operation can reach the Input Data Table at any point during the processor's pass through the ladder logic rungs, the force operation overrides the toggle operation - and so a ZERO is always stored in the bit/box. The XIC at Position H always sees the forced ZERO status of the Start_Button signal - and so the XIC gets executed as FALSE. The Pump's output bit stays in its ZERO status - and that means that the Horn will NOT sound.

Beyond Beginner Quiz #215 - Discussion

While the following material isn't intended to be a full "lesson" on all of the concepts involved, here are some basic ideas that might be helpful in understanding the results of the quiz.

First of all, the "Toggle Bit" feature simply reverses the status of a "bit" or, in other words, a "box" located in the processor's memory. Specifically, we are NOT toggling any of the instructions (XIC, XIO, OTE, etc.) located in the ladder logic. With that in mind, and referring to Figure 2, it should be obvious that toggling at (for example) Position B (an OTE) will always give EXACTLY the same results as toggling at Position D (an XIC) as long as both instructions share exactly the same address.

Second, the "Force" feature does NOT directly act upon bits (boxes) located in the processor's memory. Instead, the force is applied to either (1) an input "signal" coming into the processor - or (2) to an output "signal" going out of the processor. Specifically, a force is NOT applied directly to a bit in the processor's memory.

By contrast, we "toggle" BITS - but we "force" input or output SIGNALS. (Big difference!)

More specifically, input forces are applied (for example) at Positions W and X in Figure 1. Notice that these positions are located UPSTREAM of the bits/boxes on the Input Data Table. On the other hand, output forces are applied (for example) at Positions Y and Z in Figure 1. Notice that these positions are located DOWNSTREAM of the bits/boxes on the Output Data Table. This concept has a profound effect on how forces act.

Although an input force is not applied DIRECTLY to a bit/box in the processor's memory, the input force WILL affect the status of the bit/box - that's because the input signal is forced UPSTREAM of the bit/box - and the forced status (OFF or ON) is brought into the bit/box when the Input Data Table is updated. Specifically, under normal circumstances, forcing an Input signal ON will cause the associated bit/box to take on a status of ONE - regardless of whether the actual field input device is ON or OFF.

On the other hand, forcing an OUTput signal either ON or OFF will NOT change the status of the associated bit/box. As a specific example, suppose that Position B in Figure 3 is forced OFF - and that the memory bit on the processor's Output Data Table contains a ZERO. A "Toggle Bit" operation performed at Position B will manually give the Pump's bit in the processor's memory a status of ONE. The fact that the Pump's device in the field happens to be forced OFF has nothing to do with the ONE or ZERO status of the memory bit. This concept helps explain why a toggling function seems to "outrank" a force in certain situations.

Now let's concentrate very briefly on why so many technicians firmly believe the old rule: "You can't toggle an input bit." This is a very common misconception. Watch what happens when an input bit is toggled and you'll see that it will immediately go right back to its original state. The processor's normal update of the Input Data Table causes this effect - and maybe that instant "change back" is what makes people think that the bit never got toggled at all. Actually it WAS toggled. The confusing part is that it just didn't STAY in that toggled state very long.

Our working definition of toggling a bit is that the bit's status will be reversed from a ZERO to a ONE - or vice versa - from a ONE to a ZERO. So can we - or can we not - "reverse" or "toggle" the status of an input bit? The simple answer is: "Yes, we certainly can." So now the question becomes: Will the status of the input bit STAY reversed after we've toggled it? And the answer to that question is: "Not usually - and not for very long." Consider that the processor keeps updating its Input Data Table at a very rapid rate. Each and every time that the update operation is performed, the actual ON or OFF status of the actual field device is rewritten into the table as either a ONE or a ZERO. And that rapid "rewriting" has the effect of quickly canceling out our "Toggle Bit" operation.

Side trip: You might have noticed that this month's **Beginner Level Quiz #115** covered toggling bits which only had OUTPUT addresses. That's because toggling OUTPUT bits is much easier for beginners to cut their teeth on. We saved toggling the more difficult INPUT bits for this month's **Beyond Beginner Quiz #215** - the one that we're discussing now.

Now let's take the basic question: "Can we toggle an input bit?" and rephrase it in a way that will be more useful to a busy technician.

"After we've toggled an input bit, will it stay toggled long enough for the ladder logic to act on the new state?" That's much more to the point.

The answer to that question depends mainly on which platform we're dealing with. For the SLC-500 platform, the answer is almost always: "No." Specifically, in normal circumstances our programming terminal can only communicate with the SLC processor JUST BEFORE the update of the Input Data Table. That means that any "Toggle Bit" operation that we perform on an input bit will be "refreshed" and overwritten just before the SLC processor starts to execute the ladder logic rungs. Keep that concept in mind and you should be able to understand why some of the quiz questions for the SLC-500 systems had different answers from those for the PLC-5 and ControlLogix systems.

Incidentally, if you're interested in making an SLC-500 system disobey its normal "can't toggle an input bit" rule, try experimenting with the SVC (Service Communications) instruction. Depending on where you position the SVC rung, it's possible to make the SLC-500 behave more like a PLC-5 system where toggling is concerned.

Now let's talk about a PLC-5 type system and see why the answers to the quiz were given as "probably this" or "probably that" - and not in simple "yes" or "no" terms. In many cases, whether or not an input toggle will actually affect the program's operation literally depends upon WHERE the rung is PHYSICALLY located in the program - near the beginning - or near the end. It also depends on how long the processor takes to scan the program. Suppose that we decide to toggle an input bit to see what effect the toggled state would have on our program. First you need to understand that our programming terminal is able to communicate with our PLC-5 processor at almost any random time DURING THE EXECUTION of the ladder logic. IF (big IF) our "Toggle Bit" communication signal just happens to occur AFTER the update of the Input Data Table - but BEFORE the XIC is evaluated in the ladder logic, then yes, the toggled state WILL affect the program. On the other hand, if the processor has already passed by the XIC before the "Toggle Bit" signal gets through, then no, the toggle will NOT affect the program's operation. Specifically, the input bit will get "refreshed" with the actual ON or OFF state of the field device before the processor gets back around to the XIC on the next pass through the ladders. So with a PLC-5 system, this "random" communication is the reason why we sometimes have to toggle an input bit numerous times before the processor will finally "catch" the toggled state. Place your bets - and spin the wheel.

As a PLC-5 program becomes longer, an instruction (for example: an XIC) which is located toward the end of the program will be more likely to respond to the toggled state of an input bit than will an instruction which is located toward the beginning. With very short programs (such as the two or three rungs used in our quiz) the PLC-5 processor will almost always respond to the toggled state of an input bit.

With a ControlLogix system the communication link is even more random than with a PLC-5. Suffice it to say that toggling an input bit in a ControlLogix system is ALWAYS a "roll the dice" situation. We won't have time to go into all of the details here - but here are some basic ideas to help you think it through for yourself.

Everything else being equal, with a ControlLogix system, toggling an input bit will stand a better chance of affecting the program when (1) the input module's RPI (Requested Packet Interval) setting is longer, and (2) the program has a shorter scan time.

And here's just one more item to cause confusion with a ControlLogix system. It's very unlikely - but still quite possible - that toggling even an OUTPUT bit might not always work as intended. Basic idea: The "Toggle Bit" signal from the RSLogix5000 programming terminal is capable of reaching the ControlLogix processor even part way through the execution of a rung. That mid-rung communication normally won't happen with a PLC-5 or an SLC-500 system.

Side-by-Side Comparison of Platforms

One of the reasons why we chose to put the questions in this quiz in the "Beyond Beginner" category was that there are several situations where the three main platforms being discussed don't all respond in quite the same way - even to identical tests. Notice that there are really only six questions involved in our quiz - and each question is repeated three times - once for each of the three major platforms: PLC-5, SLC-500, and ControlLogix. The charts below should help you identify where the various platforms behaved the same - and where they behaved differently.

The fact that all Allen-Bradley platforms do NOT always function in exactly the same way is one reason why many technicians have problems when forced to work on one platform - and then switch over to another.

Question	A	B	C	D
1 - PLC	X			
2 - SLC	X			
3 - CLX	X			

Question	A	B	C	D
4 - PLC	X			
5 - SLC			X	
6 - CLX	X			

Question	A	B	C	D
7 - PLC	X			
8 - SLC			X	
9 - CLX	X			

Question	A	B	C	D
10 - PLC	X			
11 - SLC	X			
12 - CLX	X			

Question	A	B	C	D
13 - PLC	X			
14 - SLC	X			
15 - CLX	X			

Question	A	B	C	D
16 - PLC	X			
17 - SLC				X
18 - CLX				X

Beyond Beginner Quiz #215 - Summing Up

When properly used, the "Toggle Bit" and "Force" features are among the most powerful troubleshooting tools that a PLC technician has at his disposal. Unfortunately most technicians have little or no understanding of exactly how these important features actually work. Even some "Top Gun" PLC technicians with years of experience often rely on "hunt and peck" troubleshooting methods which are based on mistakes and misconceptions.

As we said before, the reason that the "wrong" answers to our quizzes seem so confusing is that they are based on the same common misconceptions that many technicians believe to be true. Unfortunately much of this material "sounds right" even though it happens to be totally wrong. The fact that these wrong ideas are so commonly believed and so widely circulated helps explain why some people find PLC skills so hard to master. If you're interested in how our PLC Boot Camp classes are specifically designed to weed out and correct these types of mistakes and misconceptions, you can find a lot of detailed information on our website at www.ronbeaufort.com.

We hope that you've enjoyed the quiz - and hopefully learned something from it. We've tried to keep the test conditions in our experiments as close as possible to the "normal" conditions that you'll run into while working in the field. But, as they say: "Your mileage may vary." Just remember that there are many factors which can affect the results of any test. That's why it's always best to learn and understand the "nuts and bolts" of why each PLC system acts the way it does - instead of relying on popular "rules of thumb" which might not be accurate in every situation.

If you'd like to discuss any of the material in our PLC Quiz, just contact us and give us a chance to go over it with you. We'll be glad to answer any questions that you might have.