**MDP VIP: Secure Cloud Manufacturing 2017 Fall**

**CNC Integration: RFID Tag Update - PLC Programming Attempts**

Date Created: 17 October 2017

**2017/10/17 Tuesday**

* RFID\_1\_JW\_ZZ: [AFI] and toggled bits
  + Edits:
    - Rung 9: Deleted [AFI]
    - Rung 9: Toggled “Robot1\_DroppedP1”, “Robot1\_DroppedP2”, “Robot1\_Flipped” bits OFF
    - Rung 21: Change from “R**3**J\_Read\_Check\_Fail” to “R**1**J\_Read\_Check\_Fail”
  + Test:
    - Check that only one “DO[#]” energies when robot places part on pallet (“DO[1]”, “DO[2]”, “~~DO[3~~]”). <- “DO[3]” should be “DO[17]”
    - “Robot1\_Flipped” energizes when part is picked up from conveyor because Rung 8 condition to energize bit is based on “DO[3]” when it should be “DO[17]” (robot DO for part is flipped)
    - Tags (for part dropped or flipped) do unlatch after “R1J\_Write\_Pass” is true, so that shouldn’t be a problem
  + Edits:
    - Rung 8: Changed from “Fanuc\_Rbt\_C1:I.Data[0].2” which is “DO[3]” to “Fanuc\_Rbt\_C1:I.Data[1].0” which is “DO[17]”
* Cell\_1\_ZZ: Current code tells Cell 1 to release the pallet after part is dropped, doesn’t wait for RFID transceiver to write to tag
  + Edits:
    - Rung 13 – “R1J\_Read\_Finished”: Changed to “R1J\_Write\_Pass”
    - Rung 13 – “Pick… Pick… RFID… CNC…”: Added [AFI] to Rung 13 to test new condition (but doesn’t work since [AFI] sets entire rung to false) <- reverted
    - Rung 13 – “Placement of part from CNC2 to conveyor Program finished DO[1]”: Added “R1J\_Write\_Pass”
  + Test:
    - Tried adding condition to release pallet after Writing command is latched, but the Write command would never execute so the pallet would stay in the cell and the robot would pick it back up again and put it in CNC 2 as if it were a new part.
    - Tags part # were being overwritten to part 0 even though I couldn’t find anywhere in the code
    - Pallet was being released before part was dropped
    - Reverted changes

**2017/11/09 Wednesday**

* Checked out most recent branch.
* Made same changes as before:
  + “RFID\_1\_JW\_ZZ”:
    - Rung 8: Changed from “Fanuc\_Rbt\_C1:I.Data[0].2” which is “DO[3]” (pick from conveyor and drop on CNC2 running) to “Fanuc\_Rbt\_C1:I.Data[1].0” which is “DO[17]” (pallet with flipped part can be let go)
    - Rung 9: Deleted [AFI]
    - Rung 9: Toggled “Robot1\_DroppedP1”, “Robot1\_DroppedP2”, “Robot1\_Flipped” bits OFF
    - Rung 21: Change from “R**3**J\_Read\_Check\_Fail” to “R**1**J\_Read\_Check\_Fail”
* To-do:
  + “RFID\_HMI”: O.Data[0] and O.Data[1] to I.Data[3] and I.Data[4] → Use AFI to avoid overwriting part #?
  + Condition for Cell 1 to release pallet?
  + “Cell\_1\_ZZ”:
    - Rung 13: Remove R1J\_Read\_Finished
    - Rung 13: Replace removed condition (R1J\_Read\_Finished) with “R1J\_Write\_Finished\_Check” OR for 3 branches (dropped from CNC1, CNC2, part flipped), add another condition checking if “R1J\_Write\_Finished\_Check”
      * Branch “place from CNC2”
      * Branch “place from CNC1”
      * Branch “part flipped”
  + “RFID\_1\_JW\_ZZ”:
    - Rung 20: Condition to unlatch part dropped from CNC1 DO[1], CNC2 DO[2], part flipped DO[17] can’t be “R1J\_Write\_Finished\_Check” or else will unlatch immediately and pallet won’t be released. Need another bit to indicate that pallet was released to unlatch DO[1]/ DO[2]/ DO[17]
* Questions:
  + “RFID\_HMI”:
    - O.Data[0] and O.Data[1] set to 0 before reading? Because it just has to be initialized to 0 before reading? This doesn’t mean it’ll write to the tag until Command is set to 14, right?
    - What is RFID\_Data1?
    - Rung 4: What is RFID\_N054:I.Channel[0].Data[2]?
    - Rung 4: What does it move from Data[7-10] to RFID\_ReadData1time[0-3]?
    - Rung 5: What is RFID\_Write\_1?? No where else in PLC routines…
  + “Cell\_1\_ZZ”:
    - Rung 12: Does it need to check that there is an empty pallet before picking up part from CNC and dropping on conveyor? **No because timer is done after pallet arrives, so if rung executes it already means there’s a pallet there**
    - Rung 13/branch 3-4 (last 2 part and process checks): What are these checking? If part 2 or 3 and process is 1? For parts 2/3, shouldn’t the pallet be released if it’s process # 1,3,4,5? In other words, only keep pallet if process # is 0 or 2, meaning it has to be machined in CNC2 or has been inspected and must be flipped? **Nevermind, this is what it already does…**
    - Rung 13/last branch: Why would it release the pallet if CNCs have part ready to be picked up?
* Edits:
  + “Cell\_1\_ZZ”:
    - Rung 13: Moved branches to after check to see if flipping, moving from CNC2, moving from CNC1
    - Rung 13: Leave R1J\_Read\_Finished
    - Rung 13: Add R1J\_Write\_Finished\_Check to each branch
* Test:
  + Empty pallet passed through normally
  + Tested with part #2, process #2 (for flipping):
    - Pallet stopped and robot flipped part, but pallet was not released (until it timed out)
    - Tag process number was correctly updated to process #3 the two times we ran the test. On the third run, the part and process number were both zero for unknown reason.
* Identified issues to resolve:
  + Since the part process is updated, “RFID\_N054:O.Channel[0].Command” must have been set to “14”, but for some reason “R1J\_Write\_Finished\_Check” is never energized or is energized for very short time that “Cell\_1\_ZZ” rung 13 checks do not match up and pallet is not released.
  + “RFID\_1\_JW\_ZZ” routine continues executing rungs that check if process number was correctly updated, and sets “RFID\_N054:O.Channel[0].Command” to “4” to re-read tag (“...Command” was switching between “0” and “4”).
  + Need to check if other tags, following read/assign/write checks are ever energized. If they are, could use those in “Cell\_1\_ZZ” rung 13 to release pallet.
* Next:
  + Need to test with other operations (CNC 1 machining and CNC 2 machining, starting with CNC 2) to see if tag process numbers are also updated correctly. Test using another tag condition on “Cell\_1\_ZZ” rung 13. Or test without “RFID\_1\_JW\_ZZ” checking if process number was successfully written (AFI).

**2017/11/12 Sunday**

* Editing branch
* Tasks:
  + AFI and other edits to revert tag bit condition
  + Test with CNC 1 and CNC 2 machining operations
  + Debug suppressed code
* To-do:
  + Rung 0: comment
    - “…tag at RFID **3**.” To “… tag at RFID **1**.”
    - “…in Cell **2**…” to “…in Cell **1**…”
    - “…has dropped a part **from CNC 1 or CNC 2**, … it will **~~also~~**... process number to **1** on the…”
    - “**If the robot in Cell 1 had flipped the part, it will write the process number to 3**
    - “…arrives at RFID **3**.” To “… arrives at RFID **1**.”
  + Rung 2: comment
    - “…RFID **3**…” to “…RFID **1**…”
  + Rung 3: comment
    - “…RFID **3**…” to “…RFID **1**…”
  + Rung 5: comment
    - “…RFID **3.**” to “…RFID **1.**”
  + Rung 10: comment
    - “…process number to 1 if part B or C; write new process number to 2 if part A.” to “…process number to 1 if from **CNC 1 or CNC 2**; write new process number to **3** if part is flipped.”
  + Rung 14: Remove “R1J\_Write\_Finished\_Check” unlatch and remove “R1J\_Read\_Check\_Start” latch
  + Rungs 15-19, 21: “AFI”
  + Rung 20: Add “R1J\_Write\_Finished\_Check” unlatch
  + Rung 21: “Robot1\_DroppedP2” Source B should be 1?
* Questions
  + “RFID\_1\_JW\_ZZ”
    - Can’t we only use “Robot1\_ Dropped” and “Robot1\_Flipped”? Why specify dropped from CNC or CNC 2 when process number is same regardless? And “Robot1\_DroppedP1” is for parts dropped form CNC 2 and “Robot1\_DroppedP2” is for dropped from CNC 1 (not intuitive?)

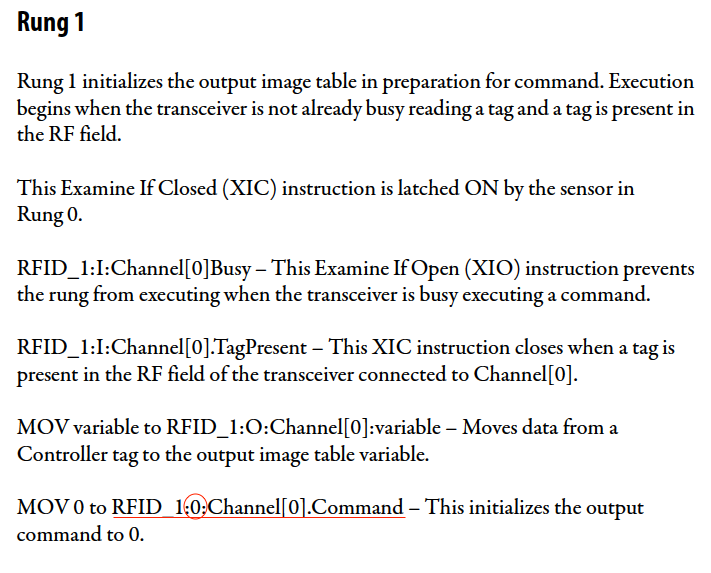
The way Ziyan programmed the routine seems to follow the example command routines and ladder logic very closely (general ladder logic used for RFID routines described in pages 68-69). But one thing I noticed from the manual is how the RFID write command (Command = 14) works.

The "Write Byte Command" (page 97) example skips the InProgress rung and instead replaces the write command with another command. This new command is set if the current "xx.O.Channel[0].Command" = 0 instead of = 14, which is what I'd assume it's supposed to be based on previous examples and the general ladder logic description that "Upon completion of the command the interface block copies the value from output command to the input command."

I thought this could explain why in "RFID\_1\_JW\_ZZ", the "...I.Channel[0].Command" tag was never 14 while we tested and why the "R1J\_Write\_Finished\_Check" never latched. Based on the example code, "...O.Channel[0].Command" is set to 14 which tells the RFID to write to the tag, but once it's executed "...I.Channel[0].Command" goes to 0 instead of 14.

But then I looked at the rung numbering in the example code on page 97 and it looks as if the image is actually either two different routines since there are two rungs numbered as 2... So I don't know if "...I.Channel[0].Command" behaves as usual and is set to 14 once the command executes (like other commands) or if it's unique for some reason.

I'll look online for other examples using the byte write command 14... I think it behaves as usual and it's likely that part of the code just got clipped out in the manual, I found something else that might've been a mistake in the manual. I'll share a screenshot. But I still don't know why when we tested it didn't change to 14 though...



**2017/11/15 Wednesday**

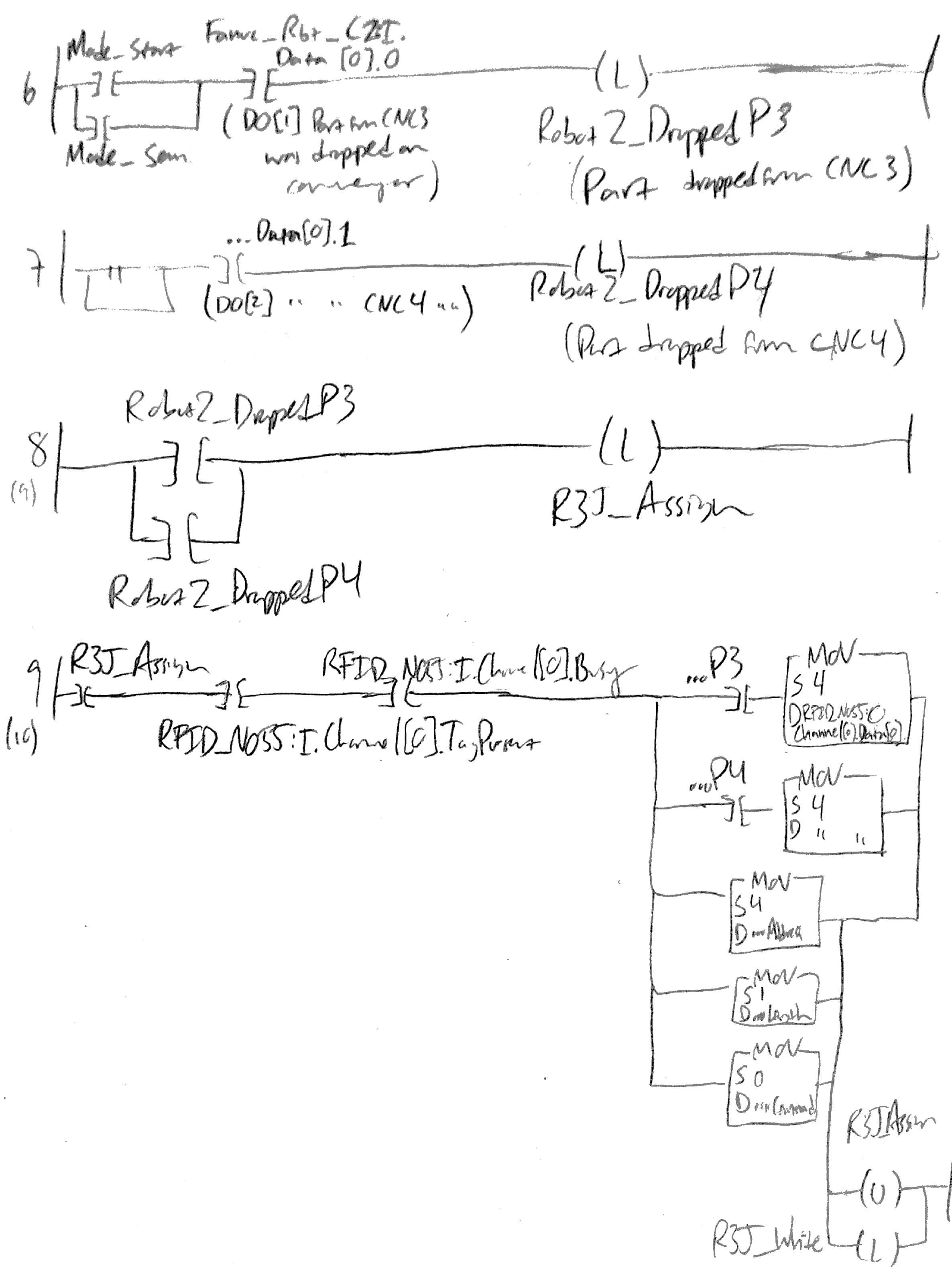
* To-do:
  + Try to use another tag for the pallet release that is latched after checking that writing command was successful ("R1J\_Write\_Pass") in the Cell 1 routine (rung 13).
  + Change the Source B for Robot1\_DroppedP2 on rung 21 from 2 to 1.
* Edits:
  + RFID\_1\_JW\_ZZ, rung 21: Change the Source B for Robot1\_DroppedP2 from 2 to 1.
  + Cell\_1\_ZZ, rung 13: Change tag for the pallet release to "R1J\_Write\_Pass" (that is latched after checking that writing command was successful)
  + RFID\_1\_JW\_ZZ, rung 13: Delete “RFID\_N054:I.Channel[0].Busy” check
  + Reverted ^
  + RFID\_HMI, rung 2: Added not “]/[“ with “R1J\_Write\_Inprogress” tag
  + Also added to:
    - RFID\_1\_JW\_ZZ: Rung 1
* Increased time that robot signal is on from 100ms to 300ms
* Need to unlatch read check fail on rung 20 after it passes the write
* Send email to Bob to notify of progress

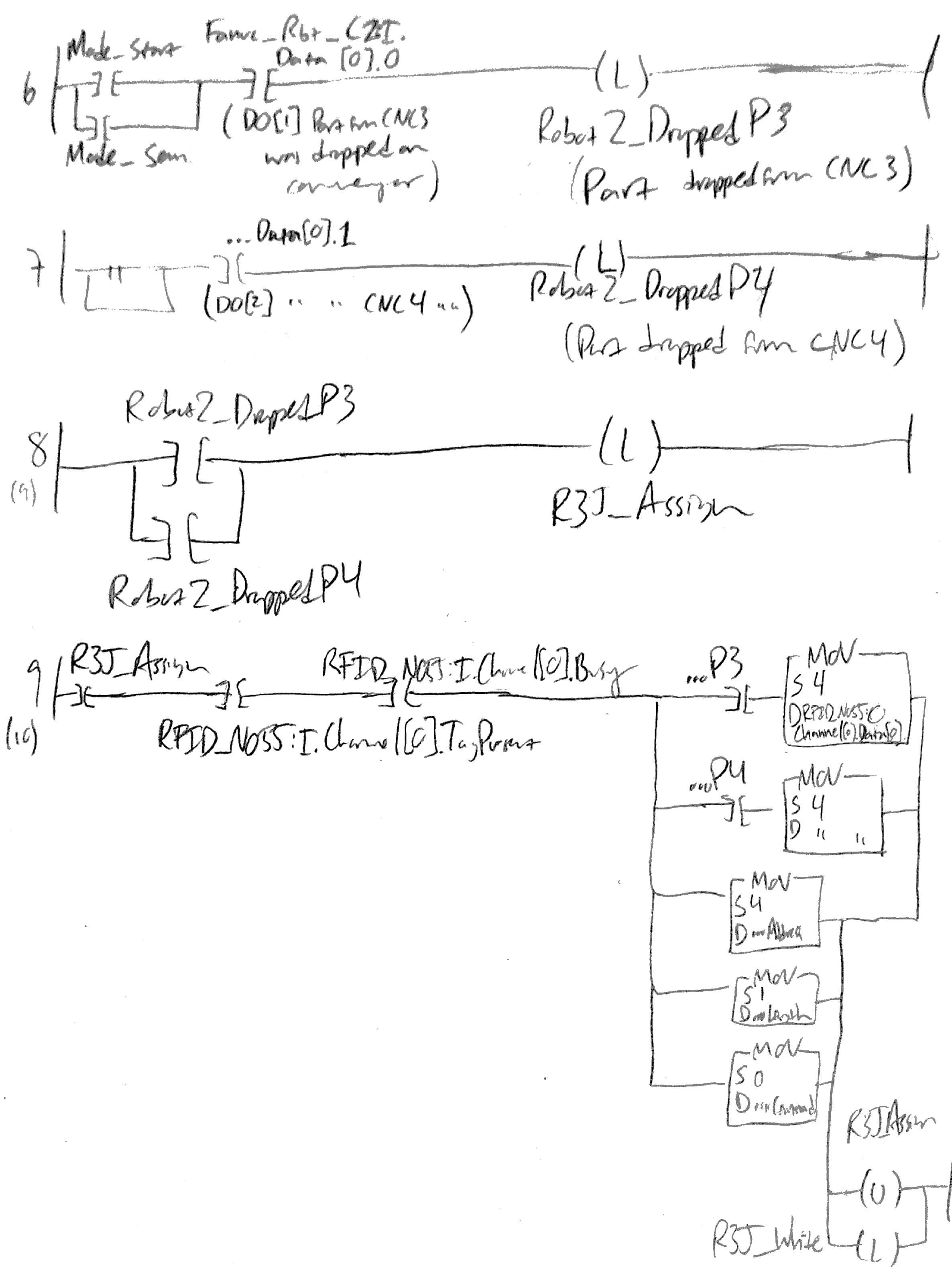
**2017/11/30 Thursday**

* To do now:
  + ~~Compile all changes made to RFID\_1\_JW\_ZZ and to Cell\_1\_ZZ to implement on main .ACD file (can’t find what it stands for, but from a~~ [~~training doc~~](http://www.automationtraining.ca/wp-content/uploads/2015/09/AB_CLX_Standard_4_Day_Module1_5C.pdf) ~~it’s the file that “contains all project databases including the program logic, data, tags, and descriptions.”)~~
  + ~~List changes to other routines, robot signals, etc. to ensure reading and writing can be accomplished~~
  + ~~Look at other routines to implement similar changes to allow for updating tags starting with Cell 2 and RFID 3~~, then inspection camera RFIDs.
* To do later:
  + Implement changes to **main** .ACD file routines
    - “Cell\_1\_ZZ”
    - “RFID\_1\_JW\_ZZ”
  + Test to make sure correct tag updating occurs after part is:
    - [1] dropped from CNC 1 (process #1→#2)
    - [2] dropped from CNC 2 (process #1→#2)
    - [3] part flipped (process #2→#3)
  + Implement changes to **branch** .ACD file routines
    - “Cell 2 routine”
    - “RFID 3 routine”
  + Test to make sure correct tag updating occurs after:
    - [1] part is dropped from CNC 3 (process #3→#4)
    - [2] part is dropped from CNC 4 (process #3→#4)

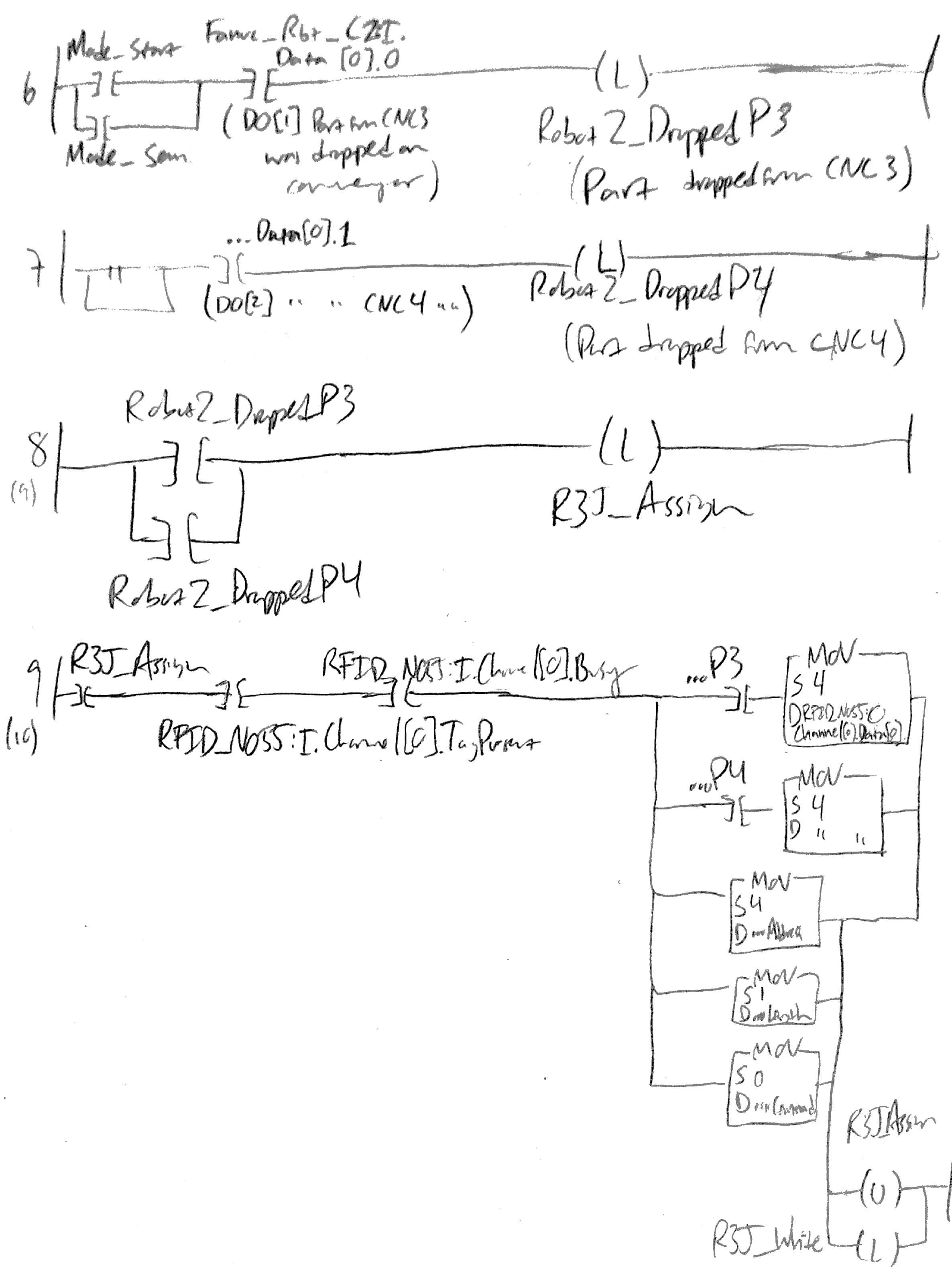
**\*\*\*\*\*\*\*\*\*\* TO-DO NOW \*\*\*\*\*\*\*\*\*\***

* All changes for “**RFID\_1\_JW\_ZZ**”
  + Rung 1: Added “]/[“ “R1J\_Write\_Inprogress”
    - To make sure the PLC doesn’t try to command RFID transceiver to read while still trying to write to RFID tag. Before this, RFID would try to read the tags immediately after writing, so rungs that had the “writing” tag would never execute since the “writing” tag would be energized for too short of a time (“writing” tag being “RFID\_N054:I.Channel[0].Command” = 14)
  + Rung 8: Changed “Fanuc\_Rbt\_C1:I.Data[0].2” to “Fanuc\_Rbt\_C1:I.Data[1].0”
    - DO[3] = “robot is in the process of moving a part to CNC2” to DO[17] = “part was flipped (pallet can be released)”
  + Rung 9: Deleted “[AFI]”
    - Was included to run demo since this code was not tested
  + Rung 9: Toggled “Robot1\_DroppedP1”, “Robot1\_DroppedP2”, “Robot1\_Flipped” bit OFF
    - Shouldn’t be energized at start
  + Rung 21: Changed from “R3J\_Read\_Check\_Fail” to “R1J\_Read\_Check\_Fail”
    - R3J is for another RFID/Cell or may not even exist; i.e., irrelevant
  + Rung 21: Changed “Robot1\_DroppedP2” branch “NEQ” “Source B” from “2” to “1”
    - If a part was machined in CNC 2 (part 3 or 4), the process # should be updated from 0 (raw material) to 1. Process #2 is for after being inspected
* All changes for “**Cell\_1\_JW\_ZZ**”
  + Rung 13: Changed branch for “R1J\_Read\_Finished” from right after DO[16] check to right after DO[6] check (before tag present check)
    - So that PLC only checks if initial read operation has taken place if the robot is not busy
  + Rung 13: In “R1J\_Read\_Finished” branch (after check) for DO[1]/DO[2]/DO[17] branches added “] [“ “R1J\_Write\_Pass” check
    - Now the pallet is only released after the RFID routine confirms that writing has taken place (tag process number is updated and correctly updated) AND robot has completed its operation (dropped from CNC 1 or 2, or flipped part). For this to occur, robot signal should last long enough for writing to take place!
* All changes for “**RFID\_HMI**”
  + Rung 2: Added “]/[“ “R1J\_Write\_Inprogress”
    - To make sure the PLC doesn’t try to command RFID transceiver to read while still trying to write to RFID tag. Before this, RFID would try to read the tags immediately after writing, so rungs that had the “writing” tag would never execute since the “writing” tag would be energized for too short of a time (“writing” tag being “RFID\_N054:I.Channel[0].Command” = 14)
* **FANUC robot 1 signals** (need to last long enough for writing and check operations to take place
  + DO[1] increased ON time from 100ms to 300ms
  + DO[2] increased ON time from 100ms to 300ms
  + DO[17] increased ON time from 100ms to 300ms
* Changes to be made on “**RFID\_3\_JW**”
  + Rung 1: Add “]/[“ “R3J\_Write\_Inprogress”
  + Rung 6&7: Latch tags for PLC use after robot in cell 2 has placed part form CNC 3 or 4 on pallet

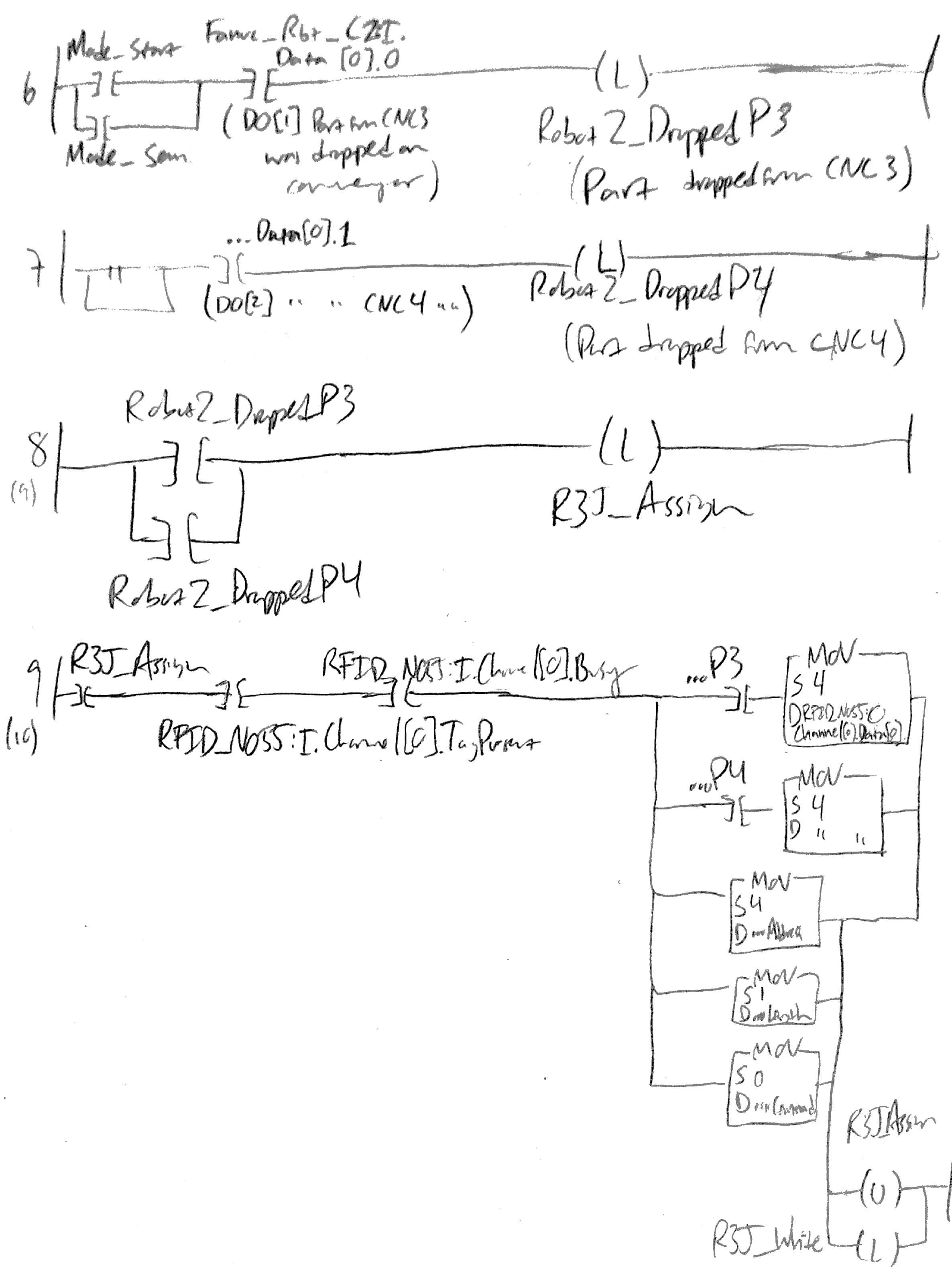




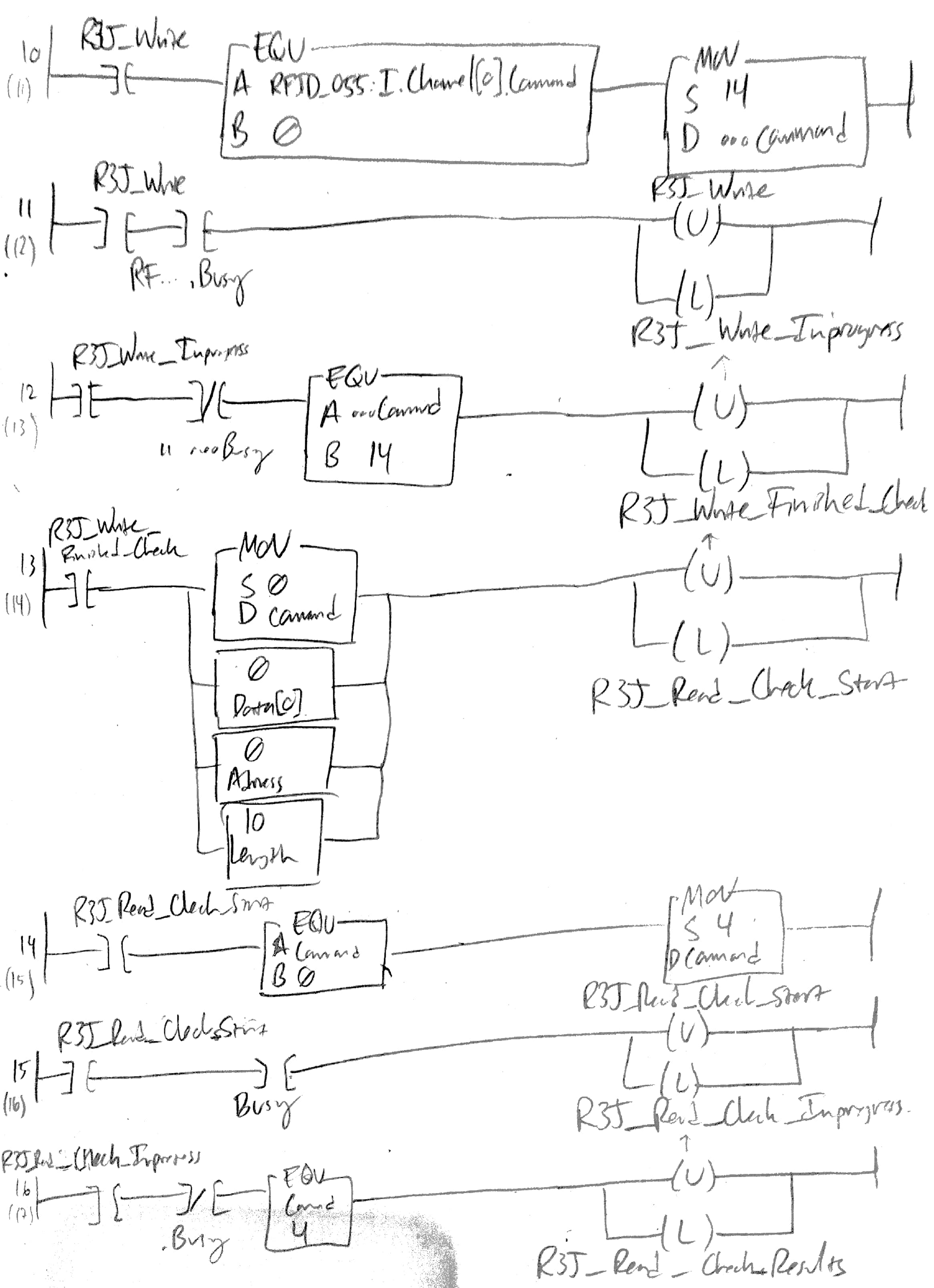
* + Rung 8: If a part has been placed on pallet after machining, begin assigning process numbers



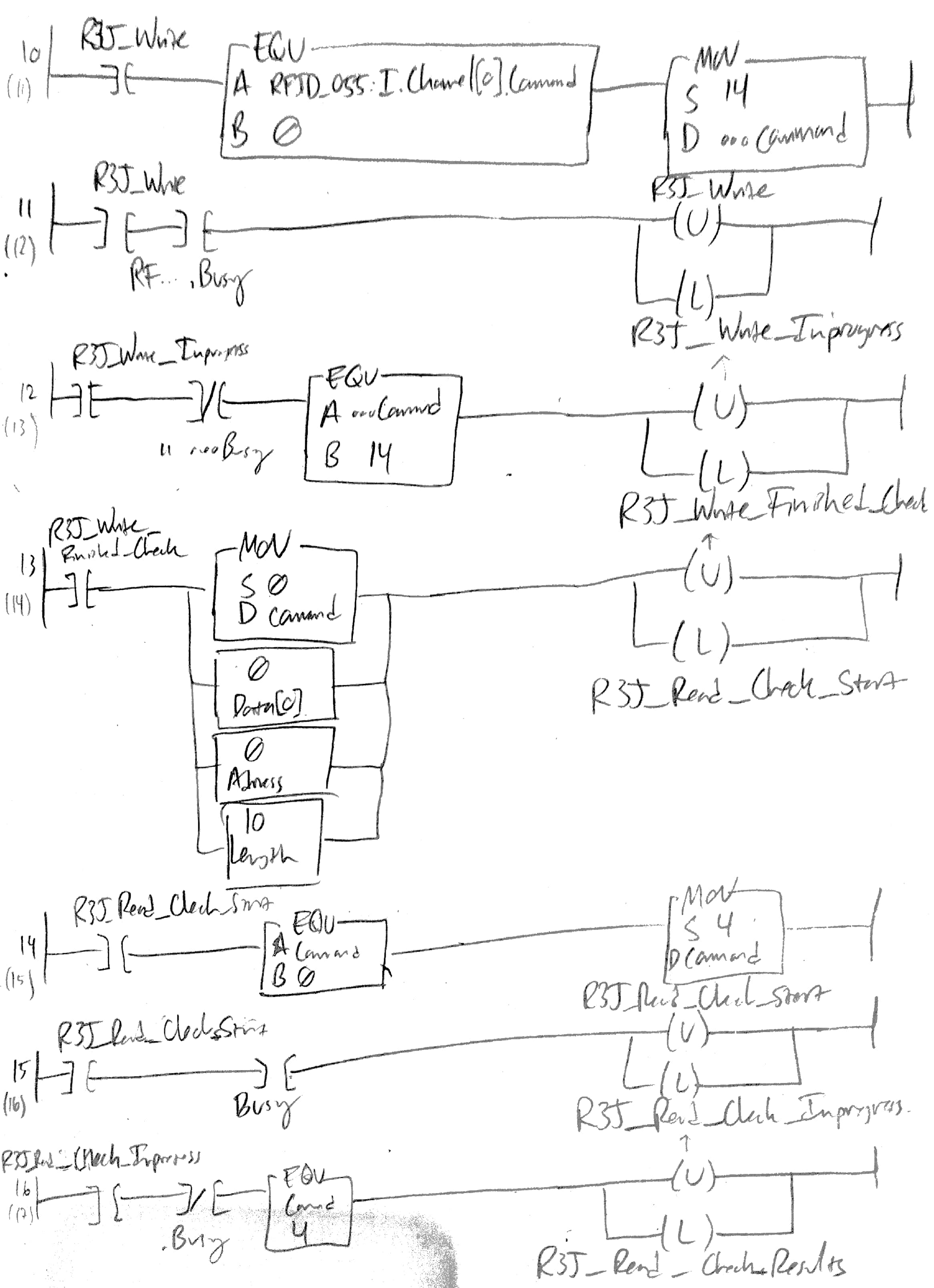
* + Rung 9: Assign process numbers and prepare RFID transceiver for writing operation. Regardless of which CNC the part was dropped from (part 2 from CNC 3 or part 3 from CNC 4), the process numbers update from 3 to 4.



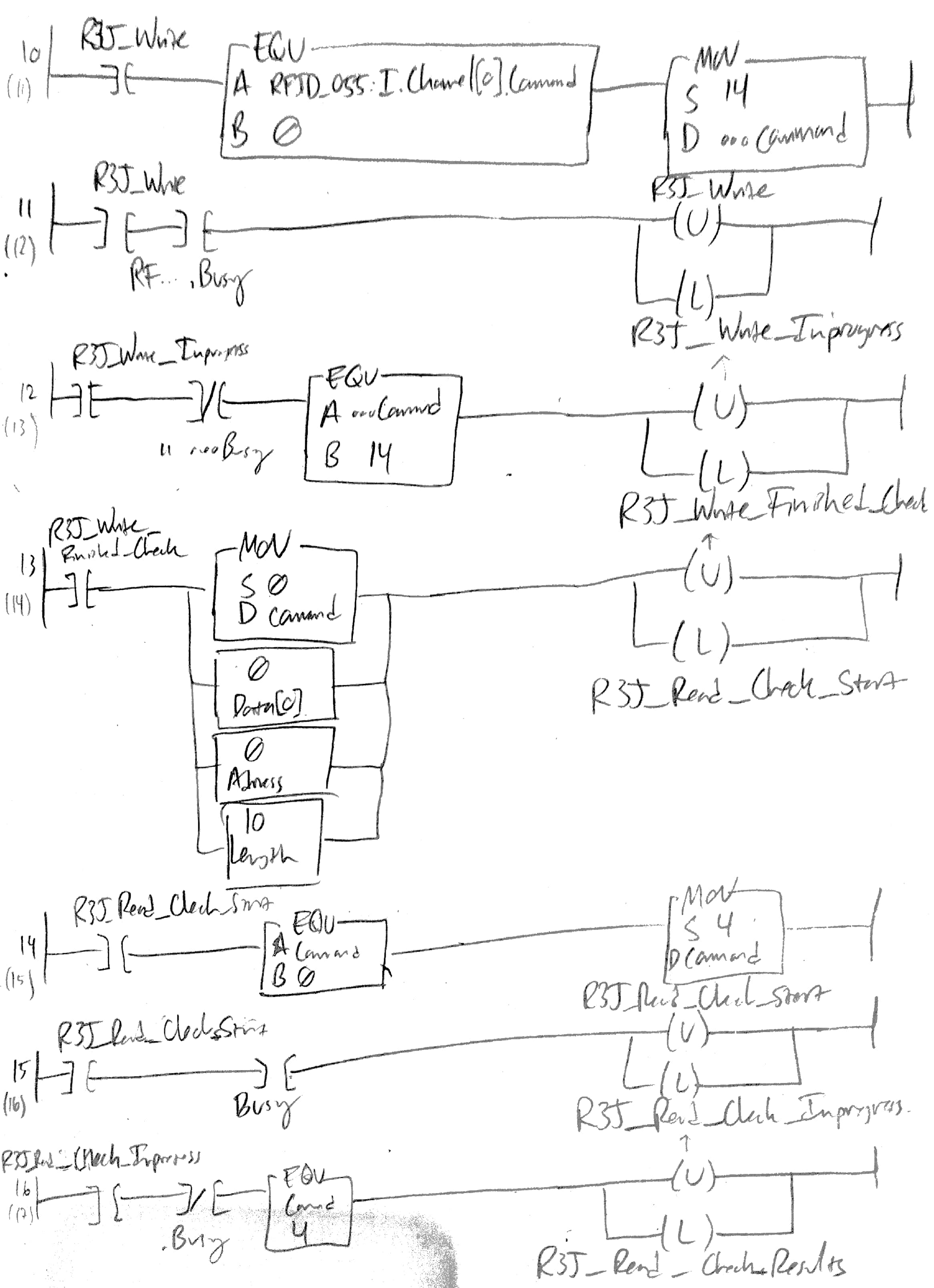
* + Rung 10: Begin writing operation



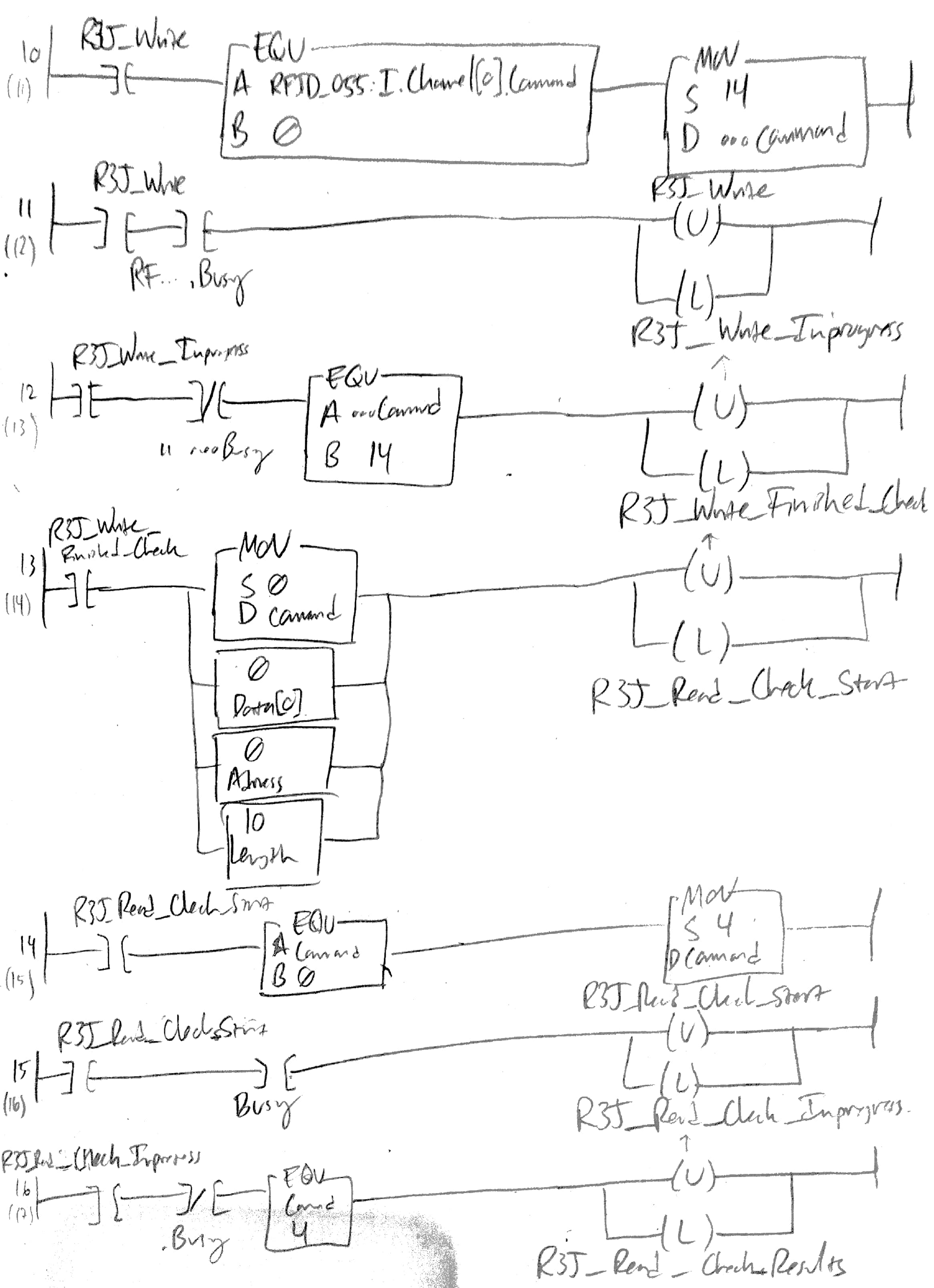
* + Rung 11: Indicate that writing is in process



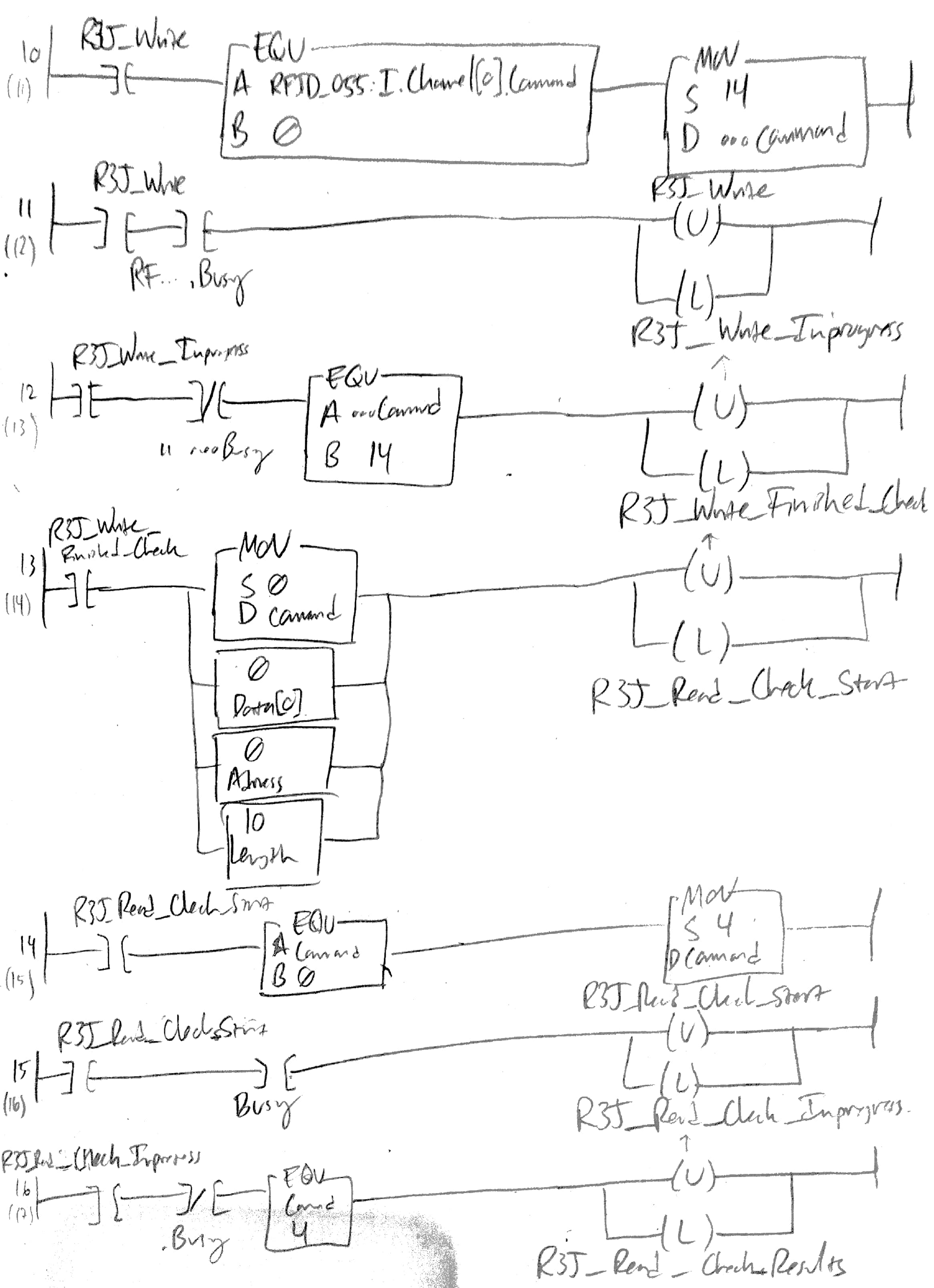
* + Rung 12: Indicate that writing operation is complete



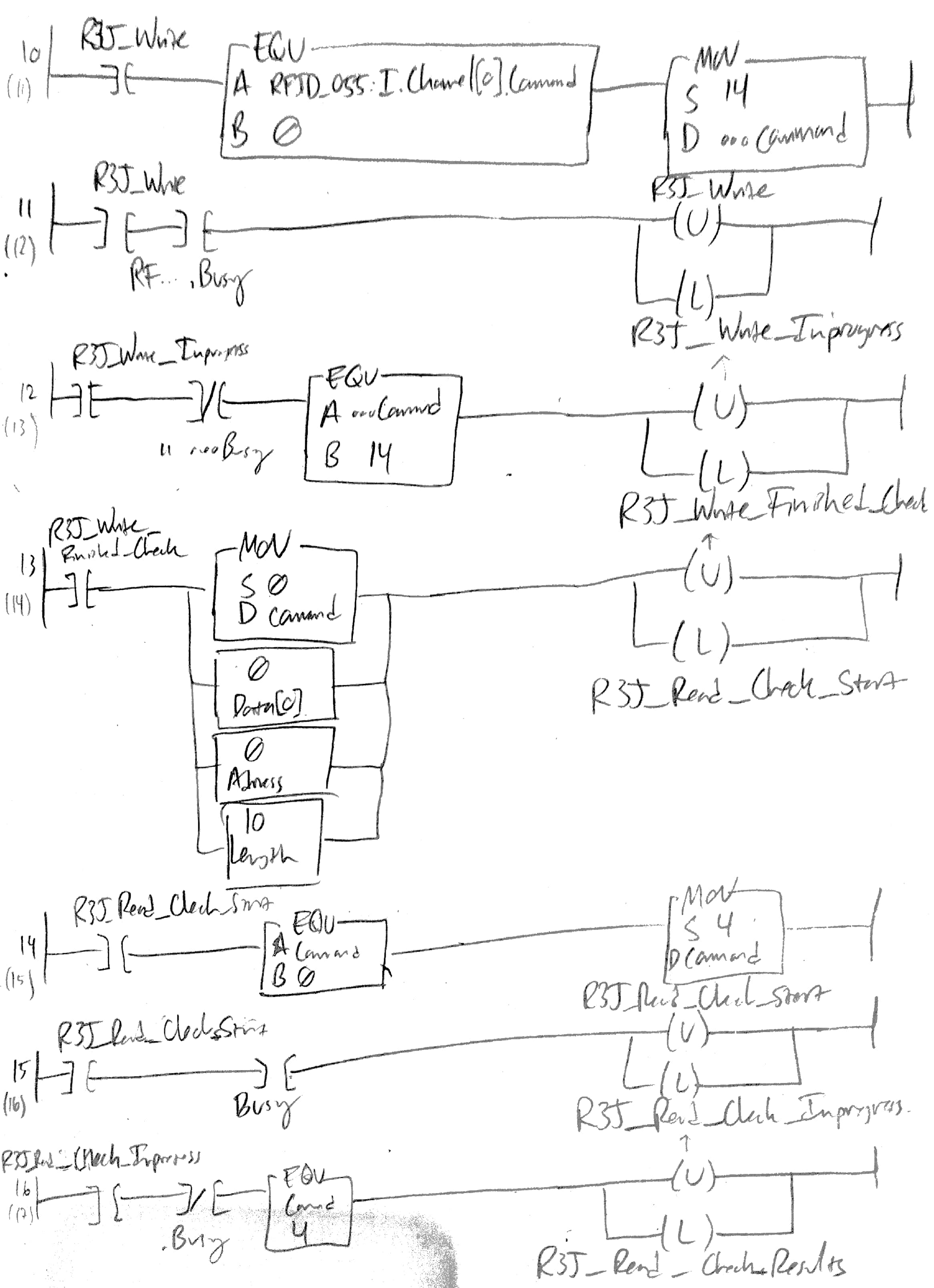
* + Rung 13: Reset RFID transceiver to begin check that tag has been correctly updated



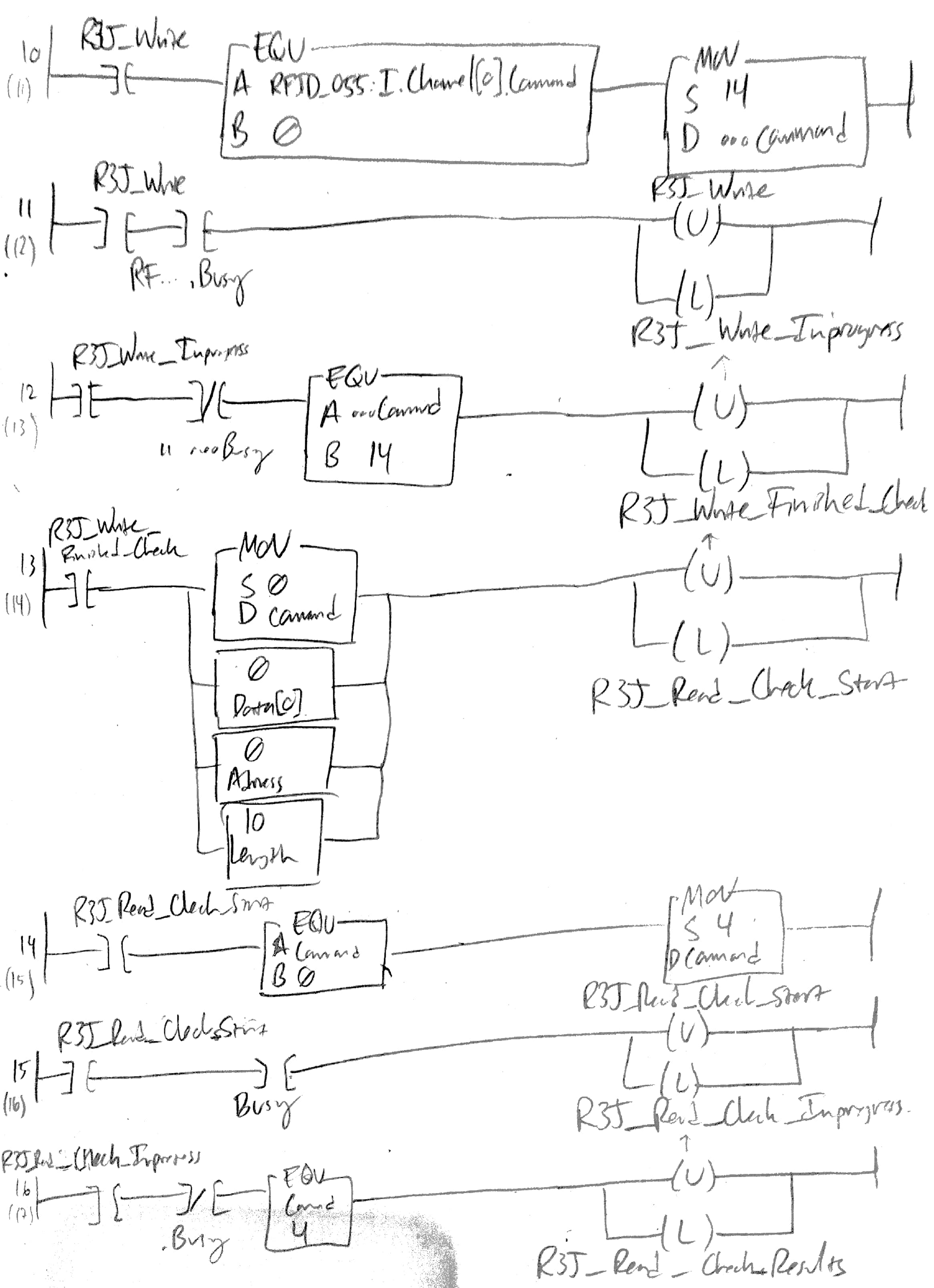
* + Rung 14: Begin read to check tag writing



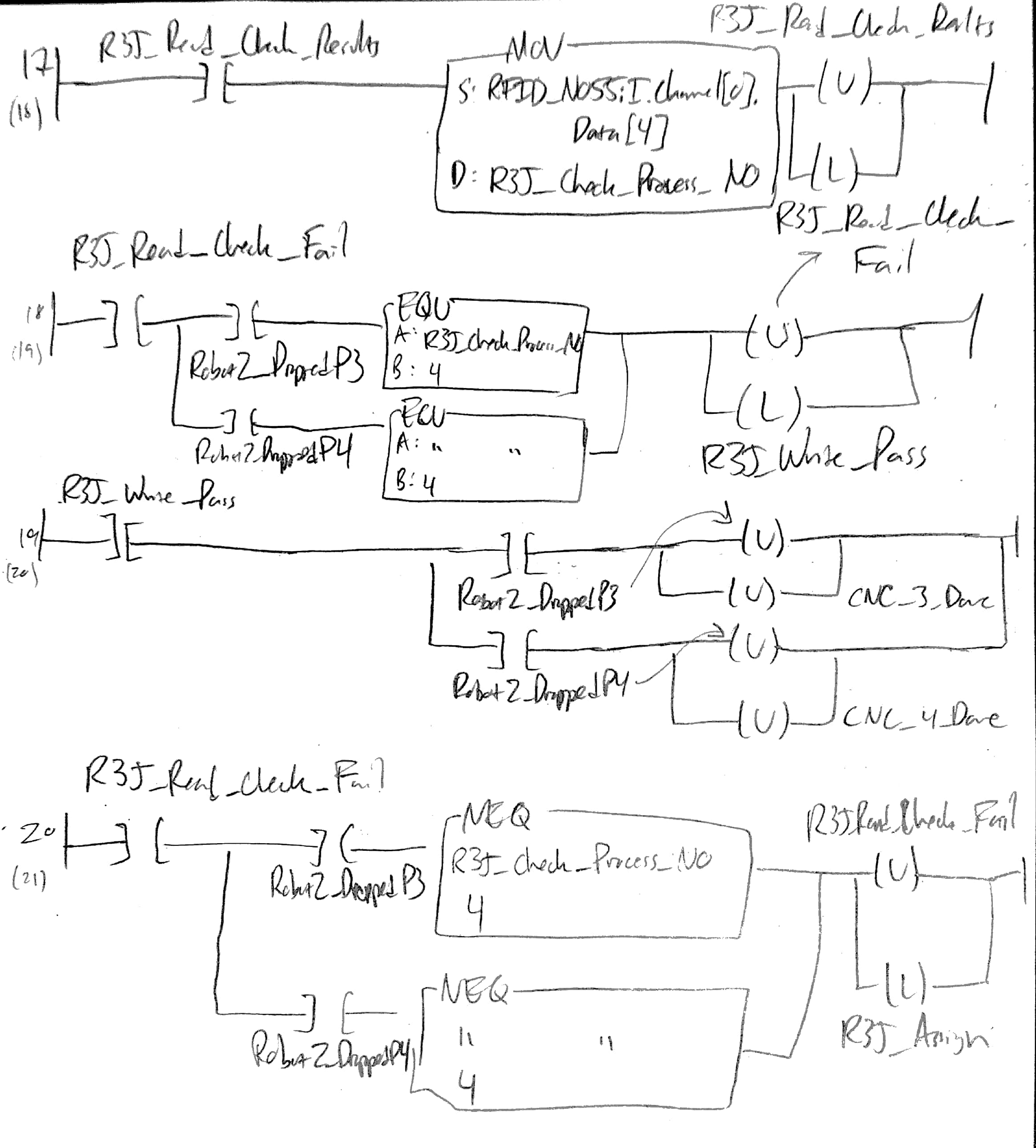
* + Rung 15: Indicate that read check is in progress



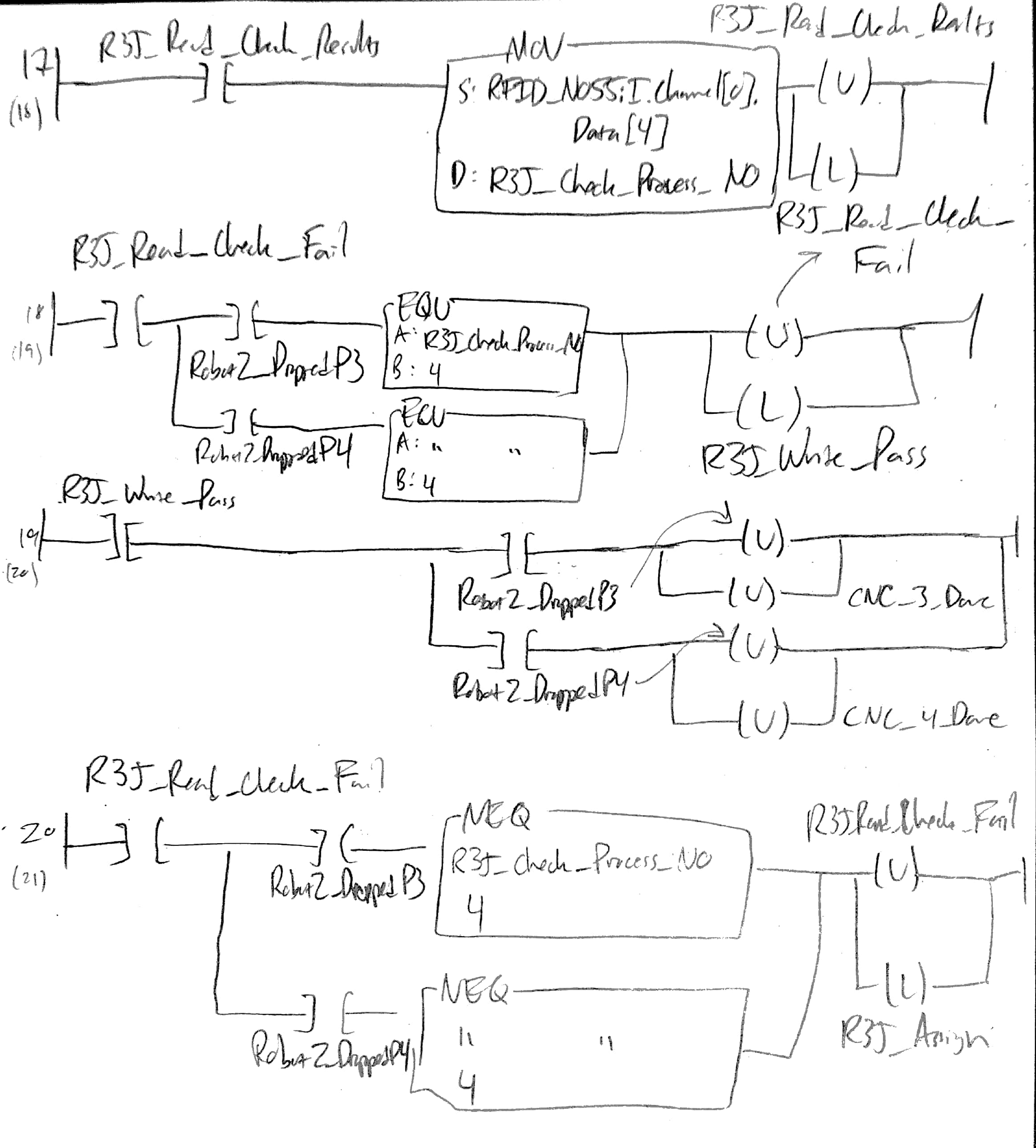
* + Rung 16: Indicate that read check operation is complete



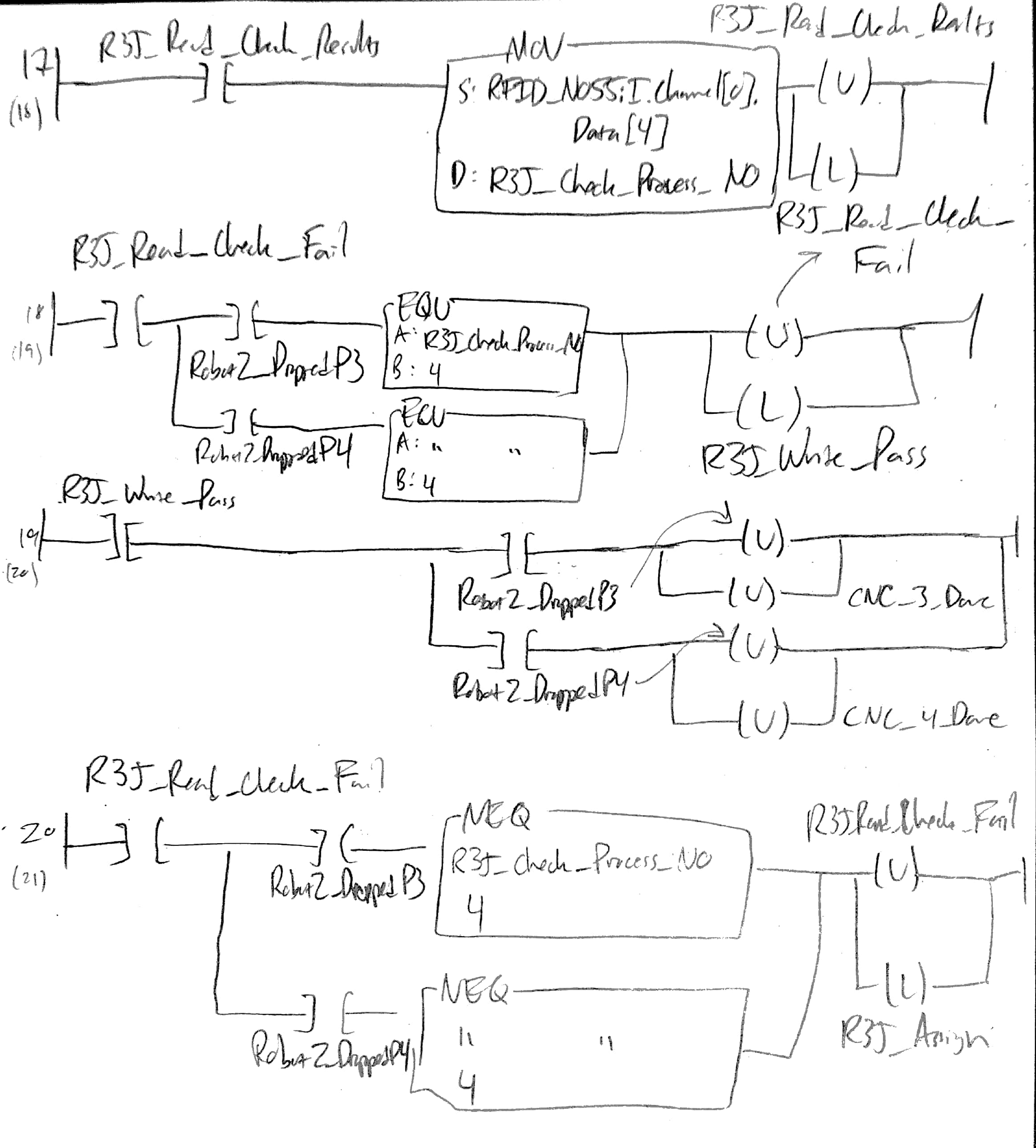
* + Rung 17: Store read process number



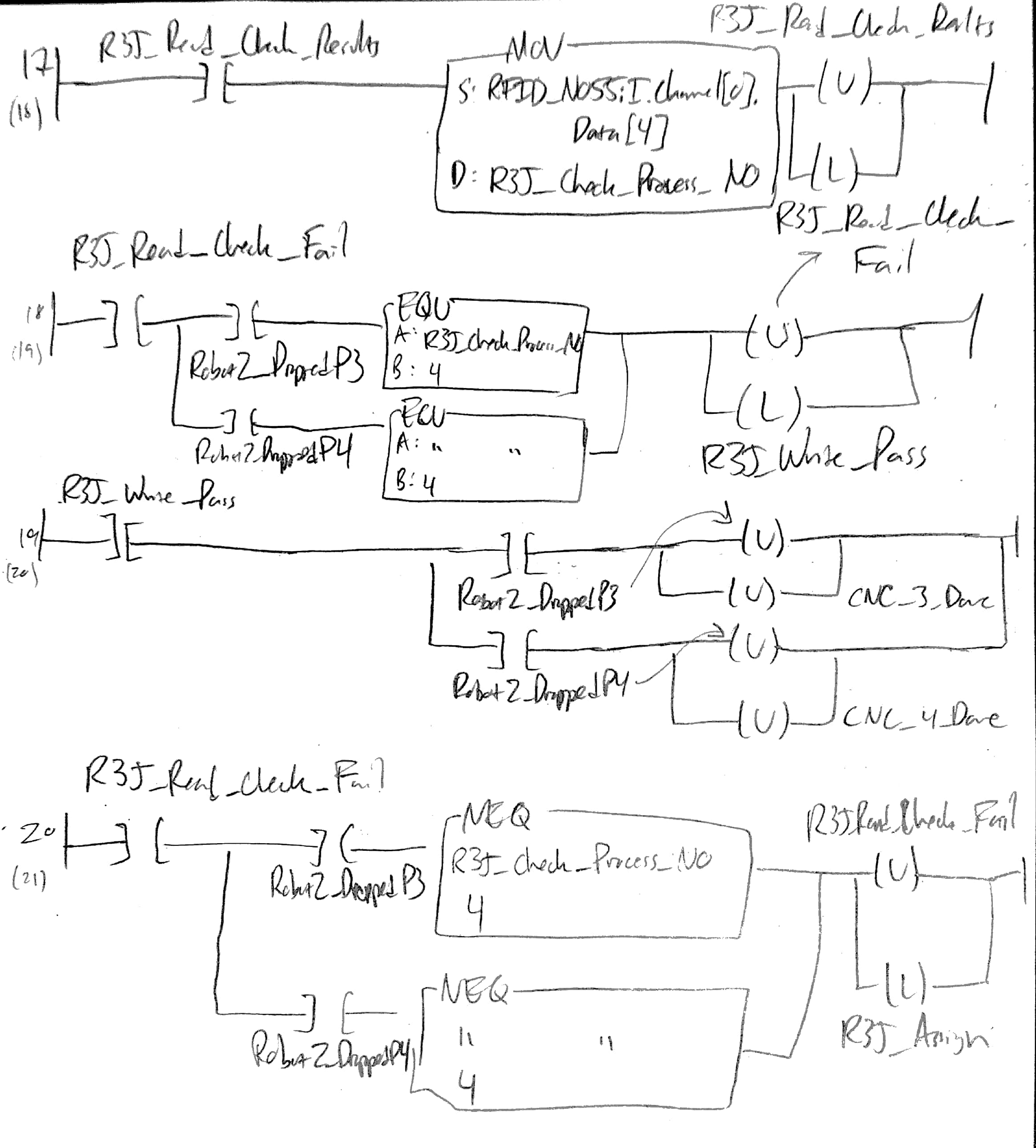
* + Rung 18: Check if process number is the correct one



* + Rung 19: If so, unlatch tags and reset CNC\_x\_Done bits



* + Rung 20: If not, return to assign rung to attempt rewrite



* Changes to be made on “**Cell\_2\_ZZ**”
  + Rung 10:
    - See if “Fanuc\_Rbt\_C2:I.Data[0].4” DO[5] and “Fanuc\_Rbt\_C2:I.Data[0].5” DO[6] exist (like in “Cell\_1\_ZZ”)
    - If so, add “]/[“ checks before branching to “R3J\_Read\_Finished”
  + Rung 10: In “R3J\_Read\_Finished” branch (after check) for DO[1]/DO[2] branches add “] [“ “R3J\_Write\_Pass” check
    - Now the pallet is only released after the RFID routine confirms that writing has taken place (tag process number is updated and correctly updated) AND robot has completed its operation (dropped from CNC 3 or 4. For this to occur, robot signal should last long enough for writing to take place!
* **FANUC robot 2 signals** (need to last long enough for writing and check operations to take place
  + DO[1] increased ON time from 100ms to 300ms
  + DO[2] increased ON time from 100ms to 300ms

**2017/12/06 Wednesday**

* All changes for “**RFID\_1\_JW\_ZZ**”
  + ~~Rung 1: Added “]/[“ “R1J\_Write\_Inprogress”~~
  + ~~Rung 8: Changed “Fanuc\_Rbt\_C1:I.Data[0].2” to “Fanuc\_Rbt\_C1:I.Data[1].0”~~
  + ~~Rung 9: Deleted “[AFI]”~~
  + ~~Rung 9: Toggled “Robot1\_DroppedP1”, “Robot1\_DroppedP2”, “Robot1\_Flipped” bit OFF~~
  + ~~Rung 21: Changed from “R3J\_Read\_Check\_Fail” to “R1J\_Read\_Check\_Fail”~~
  + ~~Rung 21: Changed “Robot1\_DroppedP2” branch “NEQ” “Source B” from “2” to “1”~~
* All changes for “**Cell\_1\_JW\_ZZ**”
  + ~~Rung 13: Changed branch for “R1J\_Read\_Finished” from right after DO[16] check to right after DO[6] check (before tag present check)~~
  + ~~Rung 13: In “R1J\_Read\_Finished” branch (after check) for DO[1]/DO[2]/DO[17] branches added “] [“ “R1J\_Write\_Pass” check~~
* All changes for “**RFID\_HMI**”
  + ~~Rung 2: Added “]/[“ “R1J\_Write\_Inprogress”~~
* **FANUC robot 1 signals** (need to last long enough for writing and check operations to take place
  + DO[1] increased ON time from 100ms to 300ms
  + DO[2] increased ON time from 100ms to 300ms
  + DO[17] increased ON time from 100ms to 300ms
* Changes to be made on “**RFID\_3\_JW**”
  + ~~Rung 1: Add “]/[“ “R3J\_Write\_Inprogress”~~
  + ~~Rung 6&7: Latch tags for PLC use after robot in cell 2 has placed part form CNC 3 or 4 on pallet~~
  + ~~Rung 8: If a part has been placed on pallet after machining, begin assigning process numbers~~
  + ~~Rung 9: Assign process numbers and prepare RFID transceiver for writing operation. Regardless of which CNC the part was dropped from (part 2 from CNC 3 or part 3 from CNC 4), the process numbers update from 3 to 4.~~
  + ~~Rung 10: Begin writing operation~~
  + ~~Rung 11: Indicate that writing is in process~~
  + ~~Rung 12: Indicate that writing operation is complete~~
  + ~~Rung 13: Reset RFID transceiver to begin check that tag has been correctly updated~~
  + ~~Rung 14: Begin read to check tag writing~~
  + ~~Rung 15: Indicate that read check is in progress~~
  + ~~Rung 16: Indicate that read check operation is complete~~
  + ~~Rung 17: Store read process number~~
  + ~~Rung 18: Check if process number is the correct one~~
  + ~~Rung 19: If so, unlatch tags and reset CNC\_x\_Done bits~~
  + ~~Rung 20: If not, return to assign rung to attempt rewrite~~
* Changes to be made on “**Cell\_2\_ZZ**”
  + Rung 10:
    - ~~See if “Fanuc\_Rbt\_C2:I.Data[0].4” DO[5] and “Fanuc\_Rbt\_C2:I.Data[0].5” DO[6] exist~~ YES, DO EXIST.
    - ~~If so, add “]/[“ checks before branching to “R3J\_Read\_Finished”~~
  + Rung 10: In “R3J\_Read\_Finished” branch (after check) for DO[1]/DO[2] branches add “] [“ “R3J\_Write\_Pass” check
* **FANUC robot 2 signals** (need to last long enough for writing/check operations to occur
  + DO[1] increased ON time from 100ms to 300ms
  + DO[2] increased ON time from 100ms to 300ms

**\*\*\*\*\*\*\*\*\*\* TESTING \*\*\*\*\*\*\*\*\*\***

* Tested and things didn’t work well…
* Starting from last 24 second mark, unit = space between every vertical gridmark
  + DroppedP1: 0.2-1.5 (single peak)
  + Assign: 0.2-1.5 (~5 peaks?)
  + Data[0].0 (to write): 0.2-1.4 (~7 peaks)
  + Read\_Check\_Start: 0.4-2.25 (3 peaks)
  + Data[3].0: Starts at 22.8 seconds to 24.9 seconds (~16 peaks or dips?)
  + Data[4].0: 0.4-2.25 (3 peaks)
  + Check\_Process\_NO: 0.75-2.5 (single peak)
* Assigned happened multiple times. Probably tried reading while assigning/writing to tag
* Tags to monitor next time:
  + **“R1J\_Read”**
  + “RFID\_N054:I.Channel[0].Busy”
  + “RFID\_N054:I.Channel[0].Command” & “RFID\_N054:O.Channel[0].Command”
  + **“Fanuc\_Rbt\_C1:I.Data[0].0” and “Robot1\_DroppedP1”**
  + “**R1J\_Assign**”
  + **“RFID\_N054:O.Channel[0].Data[0]”**
  + **“R1J\_Write”**
  + **“RFID\_N054:I.Channel[0].Data[4]”**
  + “R1J\_Read\_Check\_Start”
  + **“R1J\_Read\_Check\_Results”**
  + **“R1J\_Check\_Process\_NO”**
  + **“R1J\_Read\_Check\_Fail”**
  + **“R1J\_Write\_Pass”**
* Make sure Part # 2, Process # 0
* Make robot signal last longer
* May need a tag to prevent from executing same rungs while writing rungs are being executed. Similar to “R1J\_Read”. Something like ]/[ “Tag\_Update\_InProgress” that is (L) with “R1J\_Assign” and is (U) with “...DroppedP1” and “...DroppedP2” tags
* Add a condition on rung 11 for ]/[ “RFID\_N054:O.Channel[0].Data[0]” EQU to 0 so that it won’t assign process number of 0 ever

**2017/12/08 Friday**

* “Tag\_Update\_InProgress”
* For testing, “robot tags” = “Robot1\_DroppedP1” / “Robot1\_DroppedP2” / “Robot1\_Flipped”
* Need to unlatch “R1J\_Read\_Check\_Fail” <- Actually, tag unlatch coil exists, but for some reason the rung does not execute because one of the conditions is false: “Robot1\_DroppedP1”/“Robot1\_DroppedP2”/“Robot1\_Flipped” unlatches, meaning the rungs up to those rungs (17, 19) execute even after “R1J\_Write\_Pass” is latched, which unlatches the “Robot1\_DroppedP1”/“Robot1\_DroppedP2”/“Robot1\_Flipped” tags)
* Test with part flipping: output is (part #, process #)
  + Branch 0: Works sometimes (2,3), doesn’t other times (0,0)
    - Moved “Tag\_Update\_InProgress” latch to rung 9 from rung 10
  + Branch 1: Works sometimes (2,3), doesn’t other times (0,3)
    - ~~Added ]/[ write pass check to last rung~~  <- didn’t solve anything
    - Added ]/[ R1J\_Write\_Pass to rung 9 check so that doesn’t assign if passed
  + Branch 2: Writing to zero….
    - Reverted to Branch 1
  + Branch 3 (Branch 1):
    - Worked (2,3)
    - “Process\_NO” said 3, (0,3);
    - Worked (2,3);
    - Went straight (0,0);
    - Stayed stuck in cell, out (0,0);
    - Worked (2,3);
    - Worked (2,3);
    - NO said 0 sometimes, 3 sometimes when worked
    - Worked (2,3);
    - Went straight (0,0): “R1J\_Read\_Check\_Fail” and “R1J\_Assign” are latched after pallet is released sometimes, meaning when a new pallet with a new part comes along, “R1J\_Assign” overwrites tag. Not sure how “R1J\_Assign” latches after robot tag unlatches
    - 0,3
    - 2,3
    - 0,0
    - 0,3
    - 2,3
    - 0,0
    - 2,3

1. 2,3 → Curr\_Part (2), Curr\_Process (2), Check\_Process\_NO (0), Read\_Check\_Fail (L), Assign (L), Write\_Inprogress (L) → Since Assign, will write process to 0
2. 0,0 → Curr\_Part (0), Curr\_Process (0), Check\_Process\_NO (0), Read\_Check\_Fail (L)
3. 2,3 → Curr\_Part (2), Curr\_Process (2), Check\_Process\_NO (0), Read\_Check\_Fail (L)
4. 2,3 → Curr\_Part (2), Curr\_Process (2), Check\_Process\_NO (0), Read\_Check\_Fail (L), Assign (L),
5. 0,0 → Curr\_Part (0), Curr\_Process (0), Check\_Process\_NO (0), Read\_Check\_Fail (L)

* Rung 9 ]/[ Tag\_Update\_InProgress
  + Branch 4:

1. Part stuck, wrote 0 and stayed
2. Part stuck, wrote 0 and stayed
3. Stuck, timed out
   * + Not commanding to write process number 3 even after assign is latched, tag present, not busy, robot1\_flipped…
     + Reverted to Branch 3 (Branch 1)
     + Added reassign branch and moved Tag\_Update\_Inprogress from rung 9 to 10
   * Branch 5:
     + Assign latches once but does not write process number for some reason…

* Summary: \*12/08 tests (little to no progress)\*
  + - * Tested using additional tag “Tag\_Update\_InProgress” to avoid reading while entire process for process number update is taking place (assign, write, read check, reassign/write/read check if necessary)
      * Works 65% of the time with part flipping, other times part exits with part # 0, process #3 or enters and is let go because it’s rewritten to part # 0, process #0.
      * Seems like “R1J\_Assign” is called even after “R1J\_Write\_Pass”, so when the pallet is let go, some of the latched tags are not unlatched since “Tag\_Present” is a condition for some of these to unlatch. And when a new pallet comes in, the routine continues from where it stopped and overwrites process number
      * Not sure why part number is ever overwritten.
      * Tried making “R1J\_Assign” not be latched constantly while the “Robot1\_Dropped1”/“Robot1\_Dropped2”/“Robot1\_Flipped” since that may be causing some problems, but nothing I did worked. “R1J\_Assign” would be called once, but the for some reason (maybe timing), the “Robot1\_Flipped” is not able to assign process number 3 to Data[0], so the write command doesn’t write it, and consequently the routine does not read\_check, meaning it does not realize process number is wrong given the robot tag (flipped in this case), meaning it does not try to reassign, so the pallet just freezes.

**2017/12/09 Saturday**

**Please document the issues you've found this semester for cell 1, including the ones that you've solved and prepare a table.**

* All changes for “**RFID\_1\_JW\_ZZ**”
  + Rung 1: Added “]/[“ “R1J\_Write\_Inprogress”
    - To make sure the PLC doesn’t try to command RFID transceiver to read while still trying to write to RFID tag. Before this, RFID would try to read the tags immediately after writing, so rungs that had the “writing” tag would never execute since the “writing” tag would be energized for too short of a time (“writing” tag being “RFID\_N054:I.Channel[0].Command” = 14)
  + Rung 8: Changed “Fanuc\_Rbt\_C1:I.Data[0].2” to “Fanuc\_Rbt\_C1:I.Data[1].0”
    - DO[3] = “robot is in the process of moving a part to CNC2” to DO[17] = “part was flipped (pallet can be released)”
  + Rung 9: Deleted “[AFI]”
    - Was included to run demo since this code was not tested
  + Rung 9: Toggled “Robot1\_DroppedP1”, “Robot1\_DroppedP2”, “Robot1\_Flipped” bit OFF
    - Shouldn’t be energized at start
  + Rung 21: Changed from “R3J\_Read\_Check\_Fail” to “R1J\_Read\_Check\_Fail”
    - R3J is for another RFID/Cell or may not even exist; i.e., irrelevant
  + Rung 21: Changed “Robot1\_DroppedP2” branch “NEQ” “Source B” from “2” to “1”
    - If a part was machined in CNC 2 (part 3 or 4), the process # should be updated from 0 (raw material) to 1. Process #2 is for after being inspected
* All changes for “**Cell\_1\_JW\_ZZ**”
  + Rung 13: Changed branch for “R1J\_Read\_Finished” from right after DO[16] check to right after DO[6] check (before tag present check)
    - So that PLC only checks if initial read operation has taken place if the robot is not busy
  + Rung 13: In “R1J\_Read\_Finished” branch (after check) for DO[1]/DO[2]/DO[17] branches added “] [“ “R1J\_Write\_Pass” check
    - Now the pallet is only released after the RFID routine confirms that writing has taken place (tag process number is updated and correctly updated) AND robot has completed its operation (dropped from CNC 1 or 2, or flipped part). For this to occur, robot signal should last long enough for writing to take place!
* All changes for “**RFID\_HMI**”
  + Rung 2: Added “]/[“ “R1J\_Write\_Inprogress”
    - To make sure the PLC doesn’t try to command RFID transceiver to read while still trying to write to RFID tag. Before this, RFID would try to read the tags immediately after writing, so rungs that had the “writing” tag would never execute since the “writing” tag would be energized for too short of a time (“writing” tag being “RFID\_N054:I.Channel[0].Command” = 14)
* **FANUC robot 1 signals** (need to last long enough for writing and check operations to take place
  + DO[1] increased ON time from 100ms to 300ms
  + DO[2] increased ON time from 100ms to 300ms
  + DO[17] increased ON time from 100ms to 300ms

Table form: The following list describes issues that have been corrected and issues that are still present with the current routine. Issue description, status, fix/edit (routine rung, external), description of solution

|  |  |  |  |
| --- | --- | --- | --- |
| **Issue Description** | **Status** | **Change** | **Solution Description** |
| PLC could be commanding multiple operations to RFID transceiver | ✓ | “RFID\_1\_JW\_ZZ” rung 1 | **Added “]/[“ “R1J\_Write\_Inprogress”:** to make sure the PLC doesn’t try to command RFID transceiver to read while still trying to write to RFID tag. Before this, RFID would try to read the tags immediately after writing, so rungs that had the “writing” tag would never execute since the “writing” tag would be energized for too short of a time (“writing” tag being “RFID\_N054:I.Channel[0].Command” = 14). |
| Tag indicating part flipping would be latched when the wrong robot signal is received (tag update would not occur) | ✓ | “RFID\_1\_JW\_ZZ” rung 8 | **Changed “Fanuc\_Rbt\_C1:I.Data[0].2” to “Fanuc\_Rbt\_C1:I.Data[1].0”:** DO[3] = “robot is in the process of moving a part to CNC2” and DO[17] = “part was flipped (pallet can be released)”. |
| For testing: Robot operation tags that determine process number to write would all be latched | ✓ | “RFID\_1\_JW\_ZZ” rung 9 | **Deleted “[AFI]”** and **toggled “Robot1\_DroppedP1”, “Robot1\_DroppedP2”, “Robot1\_Flipped” bit OFF:** [AFI] was included to run demo since this code was not tested and tags shouldn’t be energized at start |
| Reach check would use the wrong tag to compare process # to correct one | ✓ | “RFID\_1\_JW\_ZZ” rung 21 | **Changed from “R3J\_Read\_Check\_Fail” to “R1J\_Read\_Check\_Fail”:** R3J is used in Cell 2, RFID 3 routine |
| Read check would use the wrong process number in comparison to determine if write was successful | ✓ | “RFID\_1\_JW\_ZZ”  rung 21 | **Changed “Robot1\_DroppedP2” branch “NEQ” “Source B” from “2” to “1”**: If a part was machined in CNC 2 (part 1 or 2), the process # should be updated from 0 (raw material) to 1. Process # of 2 is for after being inspected. |
| Optimization | ✓ | “Cell\_1\_ZZ”  rung 13 | **Changed branch for “R1J\_Read\_Finished” from right after DO[16] check to right after DO[6] check (before tag present check):** So that PLC only checks if initial read operation has taken place if the robot is not busy |
| For testing: Cell 1 would not check to see if tag process number was written | ✓ | “Cell\_1\_ZZ”  rung 13 | **In “R1J\_Read\_Finished” branch (after check) for DO[1]/DO[2]/DO[17] branches added “] [“ “R1J\_Write\_Pass” check:** Now the pallet is only released after the RFID routine confirms that writing has taken place (tag process number is updated and correctly updated) AND robot has completed its operation (dropped from CNC 1 or 2, or flipped part). |
| Cell 1 would not release pallet after tag process number was updated | ✓ | Robot 1 DO signal duration | **FANUC robot 1 digital output signals (need to last long enough for writing and check operations to take place):** DO[1,2,17] increased ON time from 100ms to 300ms |
| RFID tag updating routine is being interrupted mid-execution after process number is updated and pallet is released | ✗ | “RFID\_1\_JW\_ZZ” | Additional condition check could be added to prevent simultaneous execution of rungs while tag update is taking place |
| RFID tag part and process numbers are overwritten to 0 after passing through Cell 1 | ✗ | “RFID\_1\_JW\_ZZ” | May be resolved with previous issue. But still need to identify what routine is able to overwrite part and process numbers and why that would ever happen |

**Explain why we wanted to start with cell 1 and then move on to cell 2 (codes are similar, relatively easy portability).**

Wanted to start with cell 1 because the flipping operation was just implemented and would be easier to test the routine with a short operation rather than moving to and from the CNCs and waiting for a test machining operation to complete. Since cell 1 and cell 2 would perform very similar operations, the code would be easy to implement in cell 2 once successfully tested in cell 1.

**Please email Bob a list of the issues identified with the code and the things you've tried and cc me.**

* + - * The attached routine works ~60% of the time, updating the process number after the flipping operation from 2 to 3 most of the times, other times overwriting part and process number to 0 after releasing pallet, or overwriting part number to 0 and updating process number to 3, or part gets stuck in cell until times out or doesn’t even leave. Issue might be that the routine is executing and when it eventually passes the write operation, the pallet is released and the routine execution is interrupted midway. Some tags would stay latched after write operation was completed and pallet released.
      * Seems like “R1J\_Assign” is called even after “R1J\_Write\_Pass”, so when the pallet is let go, some of the latched tags are not unlatched since “Tag\_Present” is a condition for some of these to unlatch. And when a new pallet comes in, the routine continues from where it stopped and overwrites process number.
      * On a separate branch, I tried making “R1J\_Assign” not be latched constantly while the “Robot1\_Dropped1”/“Robot1\_Dropped2”/“Robot1\_Flipped” are latched since that may be causing some problems, but nothing I did worked. Tried using a ]/[ “Tag\_Update\_InProgress” condition on the rung where “R1J\_Assign” and on the same rung the “Tag\_Update\_InProgress” is latched so that it is called once. In order to make sure the process number is written, I added a “Reassign” tag (for if the process number is not correct) to the last rung where it does that check. But “R1J\_Assign” would be called once, and for some reason (maybe timing), the “Robot1\_Flipped” is not able to assign process number 3 to Data[0], so the write command doesn’t write it, and consequently the routine does not read\_check, meaning it does not realize process number is wrong given the robot tag (flipped in this case), meaning it does not try to reassign, so the pallet just freezes.
      * Part number is also overwritten to 0 and I’m not sure where/why that happens.