

Programming and Design

2023-2024 VEXcode VR Skills Challenge (Middle School)

Millie & David

Team Name: Burning Brain

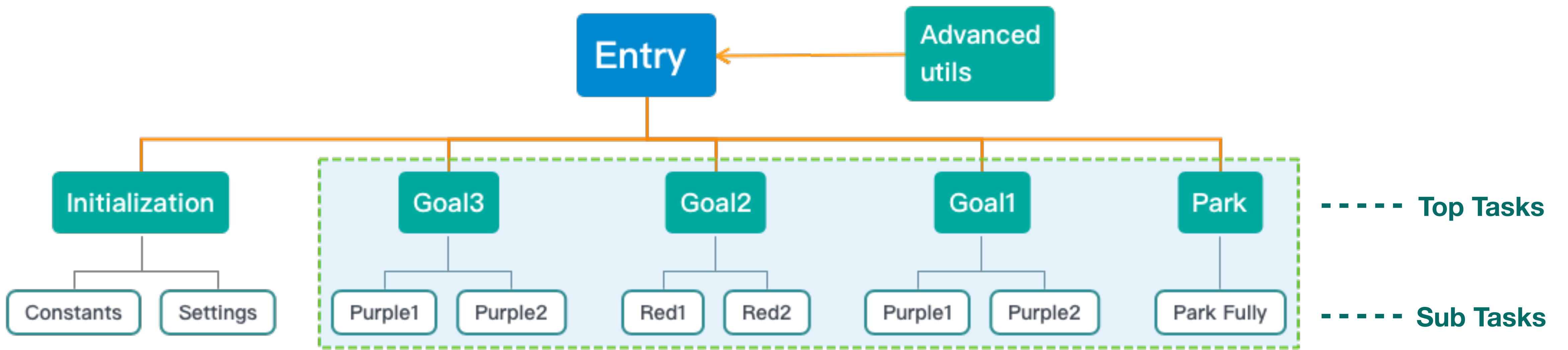
Team Number: 80066B

ChangSha, China

Table of contents

- Architecture
- Variables
- Entry
- Tasks
- Advanced Utils
- ID Pattern and Upgrade
- References

Architecture



- Drivetrain
- Motion
- Looks
- Events
- Control
- Sensing
- Operators
- Variables
- My Blocks
- Comments

```

define Park
  DEBUG start point B
  to park fully
  roll to pos armOverCube wait true
  turn right for 180 degrees
  to shake down byte by rolling its arm
  roll to pos 150 wait false
  else reverse for 11.9 inches
  waitArmComplete
  roll to pos 180 wait true
  roll to pos 0 wait true
  
```

Park

```

define GoalPurple
  DEBUG start point B
  to get purple
  GoalPurple1
  GoalPurple2
  GoalPurple3
  
```

```

define GoalPurple1
  init the purple block
  else reverse for 35 inches
  roll to pos armOverCube wait false
  turn left for 90 degrees
  else reverse for 55 inches
  turn left for 90 degrees
  roll to pos armOverCube wait true
  initBlock
  drop the purple block
  roll to pos armOverCube wait false
  else forward for 11.5 inches
  turn right for 120 degrees
  else forward for 37.4 inches
  turn left for 180 degrees
  roll to pos armReverseDrop wait false
  else reverse for 13.02 inches
  waitArmComplete
  dropBlock
  
```

```

define GoalPurple2
  init the purple block
  roll to pos armOverCube wait false
  else forward for 12 inches
  waitArmComplete
  else forward for 1.4 inches
  initBlock
  drop the purple block
  roll to pos armReverseDrop wait false
  else reverse for 13.8 inches
  waitArmComplete
  wait 0.2 seconds
  dropBlock
  
```

Goal1

```

define GoalRed
  GoalRed1
  DEBUG start point B
  to get red
  GoalRed2
  
```

```

define GoalRed1
  init the red block
  roll to pos armOverCube wait true
  else reverse for 5 inches
  turn left for 90 degrees
  else reverse for 11 inches
  turn right for 90 degrees
  waitArmComplete
  wait 0.1 seconds
  initBlock
  drop the red block
  else forward for 6 inches
  turn right for 90 degrees
  else forward for 24 inches
  turn right for 90 degrees
  else forward for 4 inches
  dropBlock
  
```

```

define GoalRed2
  init the red block
  else reverse for 12 inches
  turn left for 90 degrees
  else reverse for 12 inches
  turn left for 90 degrees
  roll to pos armOverCube wait true
  initBlock
  drop the red block
  roll to pos armOverCube wait false
  turn right for 90 degrees
  else forward for 12 inches
  turn right for 90 degrees
  else forward for 24 inches
  dropBlock
  
```

Goal2

```

define GoalPurple
  to collect 2 purple blocks and touch level 2
  GoalPurple1
  GoalPurple2
  
```

```

define GoalPurple1
  init the purple block
  roll to pos armOverCube wait false
  else forward for 33.5 inches
  initBlock
  drop the purple block
  roll to pos armOverCube wait false
  else forward for 11 inches
  turn left for 90 degrees
  else forward for 6 inches
  turn right for 90 degrees
  else forward for 24 inches
  waitArmComplete
  turn right for 90 degrees
  roll to pos armReverseDrop wait false
  else forward for 2.4 inches
  debug 0 -> 5.4
  dropBlock
  
```

```

define GoalPurple2
  init the purple block
  roll to pos armOverCube wait true
  roll to pos armOverCube wait false
  else reverse for 11 inches
  initBlock
  drop the purple block
  roll to pos armReverseDrop wait false
  else forward for 11 inches
  waitArmComplete
  dropBlock
  
```

Goal3

```

define Initialization
  InitializeConstants
  InitializeByte
  
```

```

define InitializeConstants
  set armOverCube to 0
  set armOverGreen to 60
  set armOverRed to 150
  set armOverDrop to 90
  set armReverseDrop to 120
  
```

```

define InitializeByte
  set drive velocity to 150 %
  set turn velocity to 180 %
  set IntakeMotorGroup velocity to 120 %
  set ArmMotorGroup velocity to 180 %
  
```

Initialization

```

when started
  start point B
  Initialize
  GoalPurple
  GoalRed
  GoalPurple
  Park
  
```

Entry

```

define rollArmToPos pos wait wait
  if wait then
    spin ArmMotorGroup to position pos degrees
  else
    spin ArmMotorGroup to position pos degrees and don't wait
  set armToPosition to pos
  
```

```

define waitArmComplete
  wait until ArmMotorGroup position is degrees = armToPosition
  
```

```

define initBlock
  spin IntakeMotorGroup to position 90 degrees
  wait until IntakeBumper pressed
  
```

```

define dropBlock
  spin IntakeMotorGroup to position 0 degrees
  wait until IntakeBumper pressed
  wait 0.5 seconds
  
```

Advanced
Utils

Variables

Constants

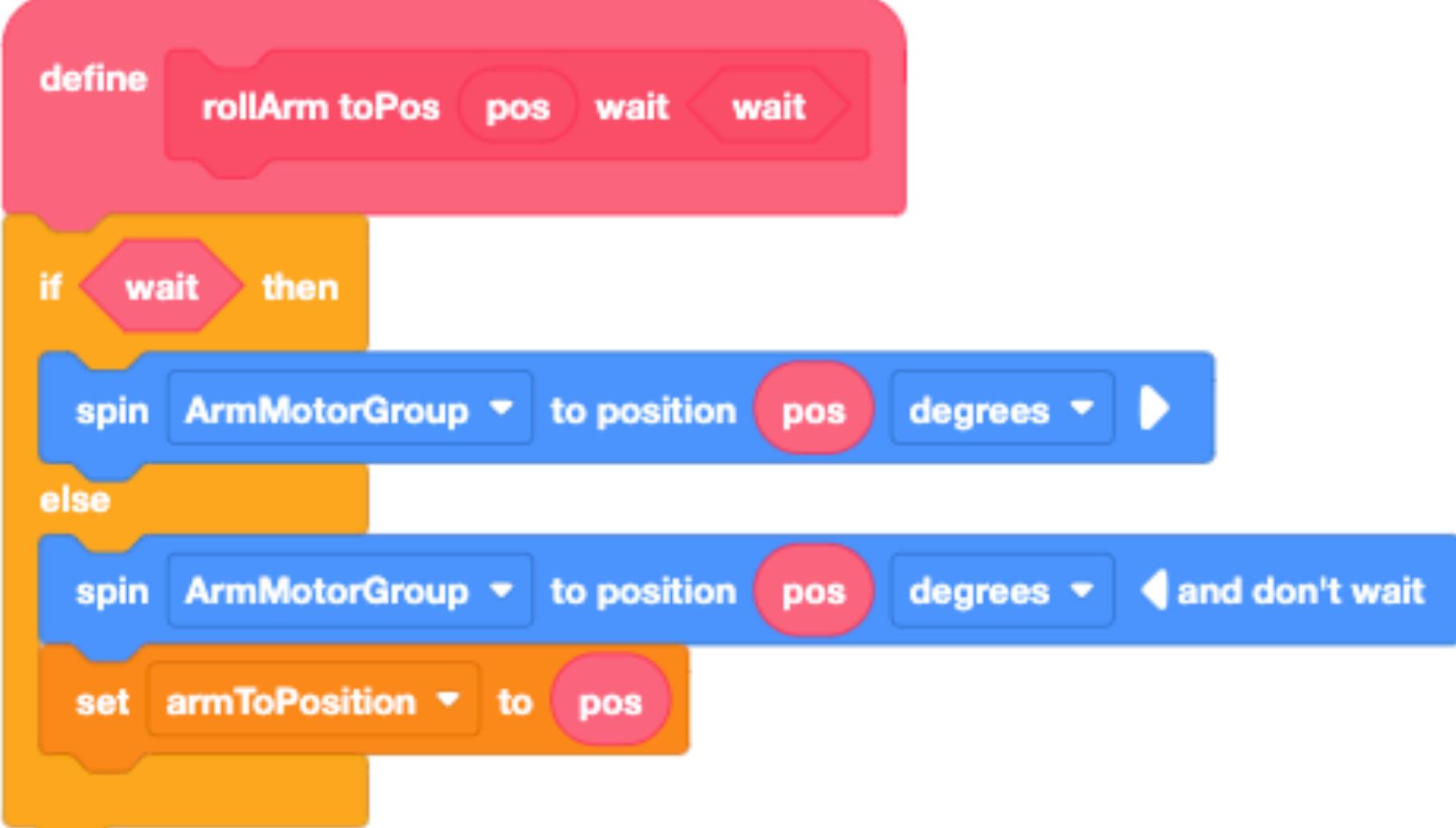
armIntake	The degrees of the arm intaking the block
armOverGreen	The degrees of the arm intaking the purple block by passing green blocks
armIntakeRed	The degrees of the arm intaking the red block
armDropRed	The degrees of the arm dropping the red block
armAvoidCube	The degrees of the arm dropping the block and avoiding touching the Goal cube
armReverseDrop	The degrees of the arm reverse dropping the block

Cross-Block Cache

- armToPosition

Using *armToPosition* variable to **cache** the expected degrees when starting to roll the arm asynchronously

Wait until the arm expected degrees is reached using the cached *armToPosition* variable



```
define rollArm toPos pos wait wait
if wait then
  spin ArmMotorGroup to position pos degrees
else
  spin ArmMotorGroup to position pos degrees and don't wait
  set armToPosition to pos
```

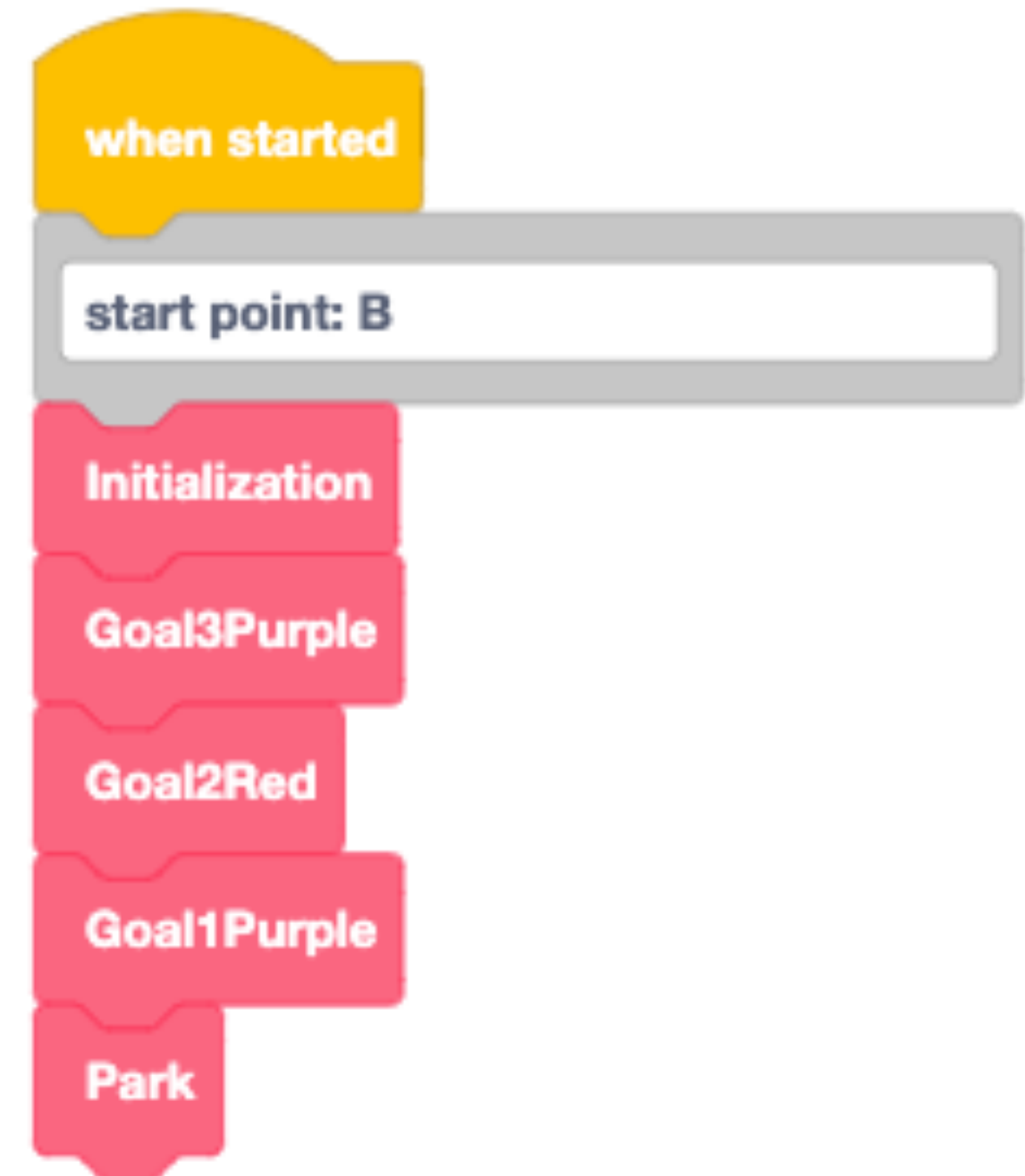


```
define waitUtilArmComplete
wait until ArmMotorGroup position in degrees = armToPosition
```


Entry

NOTE: Please follow the comments to set the startup location correctly

The entry blocks only executes **top-level tasks** sequentially and functions are implemented in task groups



Tasks

Initialization

To implement the initialization of constants and robot parameters

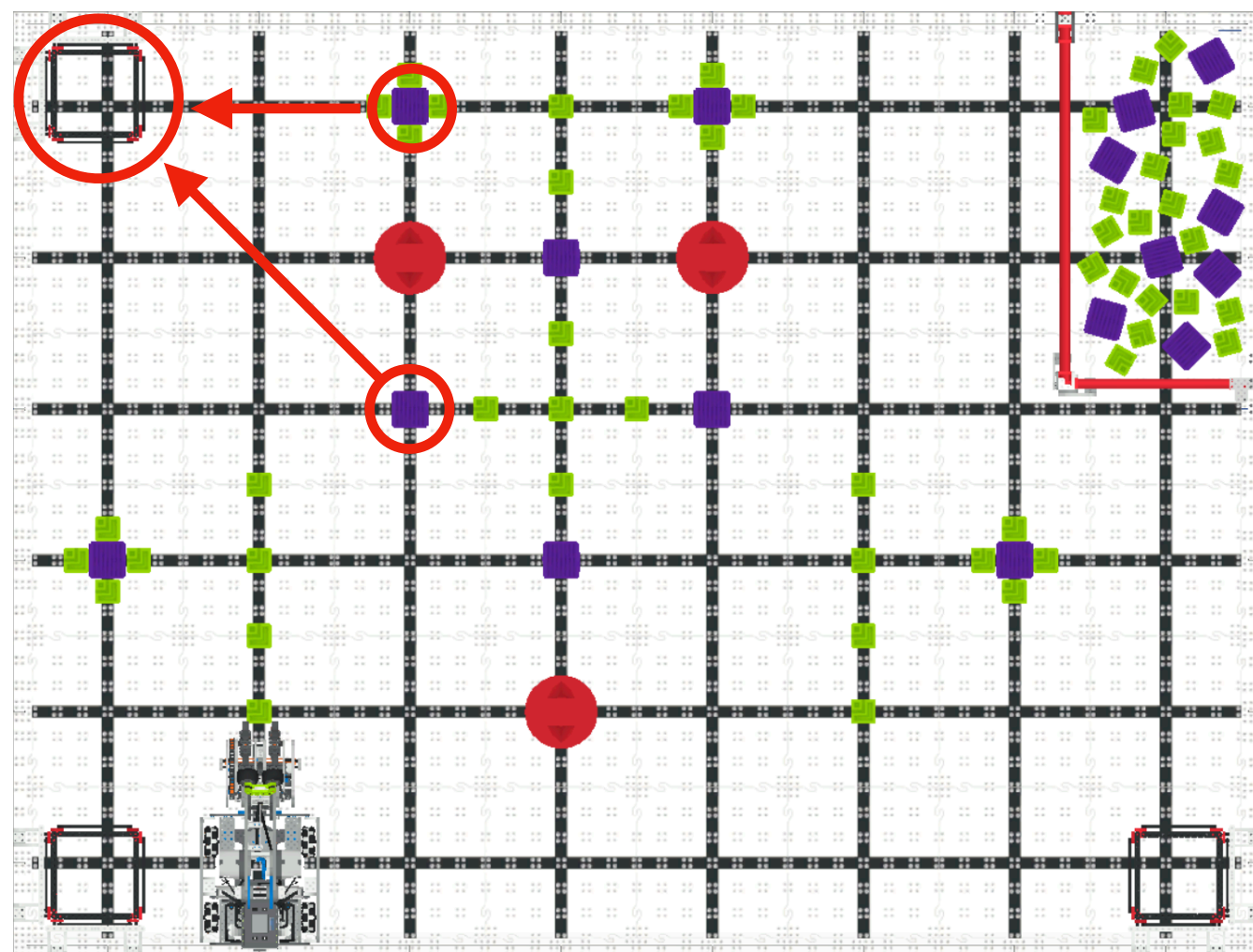
```
define Initialization
  InitializeConstants
  InitializeByte
```

```
define InitializeConstants
  set armIntake to 0
  set armOverGreen to 65
  set armIntakeRed to 110
  set armDropRed to 305
  set armAvoidCube to 985
  set armReverseDrop to 1200
```

```
define InitializeByte
  set drive velocity to 100 %
  set turn velocity to 100 %
  set IntakeMotorGroup velocity to 100 %
  set ArmMotorGroup velocity to 100 %
```

Goal 3

Collect 2 purple blocks to Goal III to achieve Uniform and Fill Level 2



```

define Goal3Purple
  to collect 2 purple blocks and touch level 2
  Goal3Purple1
  Goal3Purple2
  
```

```

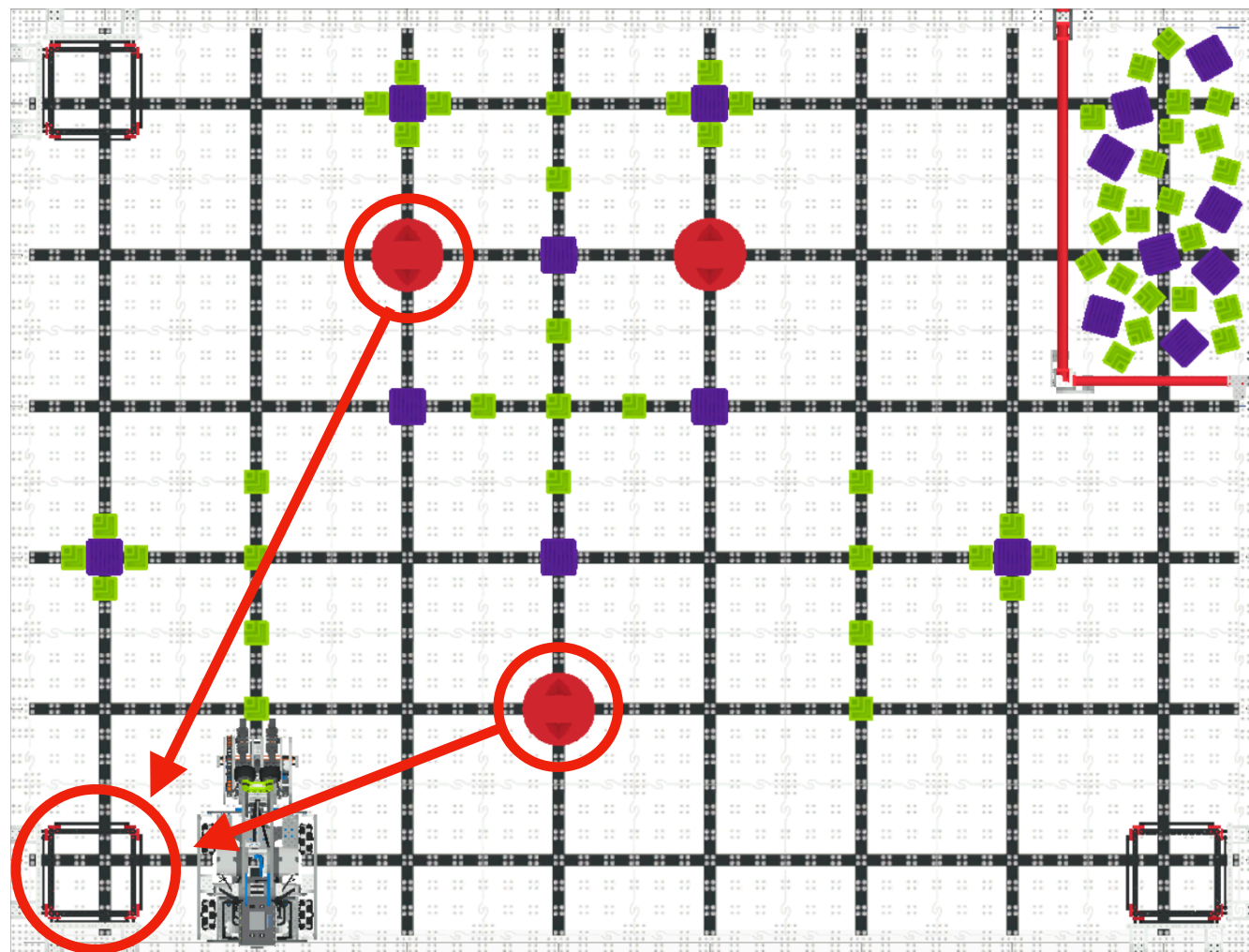
define Goal3Purple1
  intake the purple block
  rollArm toPos armOverGreen wait false
  drive forward for 23.5 inches
  intakeBlock
  drop the purple block
  rollArm toPos armAvoidCube wait false
  drive forward for 11 inches
  turn left for 90 degrees
  drive forward for 9 inches
  turn right for 90 degrees
  drive forward for 24 inches
  waitUtilArmComplete
  turn right for 90 degrees
  rollArm toPos armReverseDrop wait false
  drive forward for 2.4 inches
  detax: 5 -> 5.4
  dropBlock
  
```

```

define Goal3Purple2
  intake the purple block
  rollArm toPos armAvoidCube wait true
  rollArm toPos armOverGreen wait false
  drive reverse for 3.9 inches
  waitUtilArmComplete
  intakeBlock
  drop the purple block
  rollArm toPos armReverseDrop wait false
  drive forward for 3.5 inches
  waitUtilArmComplete
  dropBlock
  
```


Goal 2

Collect 2 red blocks to Goal 1 to achieve Uniform and Fill Level 2



```
define Goal2Red
  Goal2Red1
  DEBUG start point: B
  B2Goal2WithRed
  Goal2Red2
```

```
define Goal2Red1
  intake the red block
  rollArm toPos armAvoidCube wait true
  rollArm toPos armIntakeRed wait false
  drive reverse for 5 inches
  turn left for 90 degrees
  drive reverse for 12 inches
  turn right for 90 degrees
  waitUtilArmComplete
  wait 0.1 seconds
  intakeBlock
  drop the red block
  drive forward for 6 inches
  turn right for 90 degrees
  rollArm toPos armDropRed wait false
  drive forward for 49 inches
  turn right for 90 degrees
  drive forward for 6 inches
  dropBlock
```

```
define B2Goal2WithRed
  from start point B to Goal 2 with the red block
  rollArm toPos armIntakeRed wait false
  drive forward for 10.5 inches
  turn right for 90 degrees
  waitUtilArmComplete
  intakeBlock
  rollArm toPos armDropRed wait false
  turn right for 90 degrees
  drive forward for 13 inches
  turn right for 90 degrees
  drive forward for 12 inches
  dropBlock
  intake & drop the red block again
  drive reverse for 12 inches
  turn left for 90 degrees
  drive reverse for 13 inches
  turn left for 90 degrees
  rollArm toPos armIntakeRed wait true
  intakeBlock
  rollArm toPos armDropRed wait false
  turn right for 90 degrees
  drive forward for 13 inches
  turn right for 90 degrees
  drive forward for 12 inches
  dropBlock
```

```
define Goal2Red2
  intake the red block
  drive reverse for 12 inches
  turn left for 90 degrees
  drive reverse for 13 inches
  turn left for 90 degrees
  rollArm toPos armIntakeRed wait true
  intakeBlock
  drop the read block
  rollArm toPos armDropRed wait false
  turn right for 90 degrees
  drive forward for 13 inches
  turn right for 90 degrees
  drive forward for 12 inches
  dropBlock
```

To simulate the end state of the last step

Goal 1

```

define Goal1Purple
  DEBUG start point: B
  B2Goal2RedDrop
  Goal1Purple1
  Goal1Purple2
  
```

```

define B2Goal2RedDrop
  from start point B to Goal 2 after dropping the red block
  rollArm toPos armDropRed wait false
  drive forward for 2.5 inches
  turn right for 180 degrees
  drive forward for 5 inches
  turn right for 90 degrees
  drive forward for 12 inches
  
```

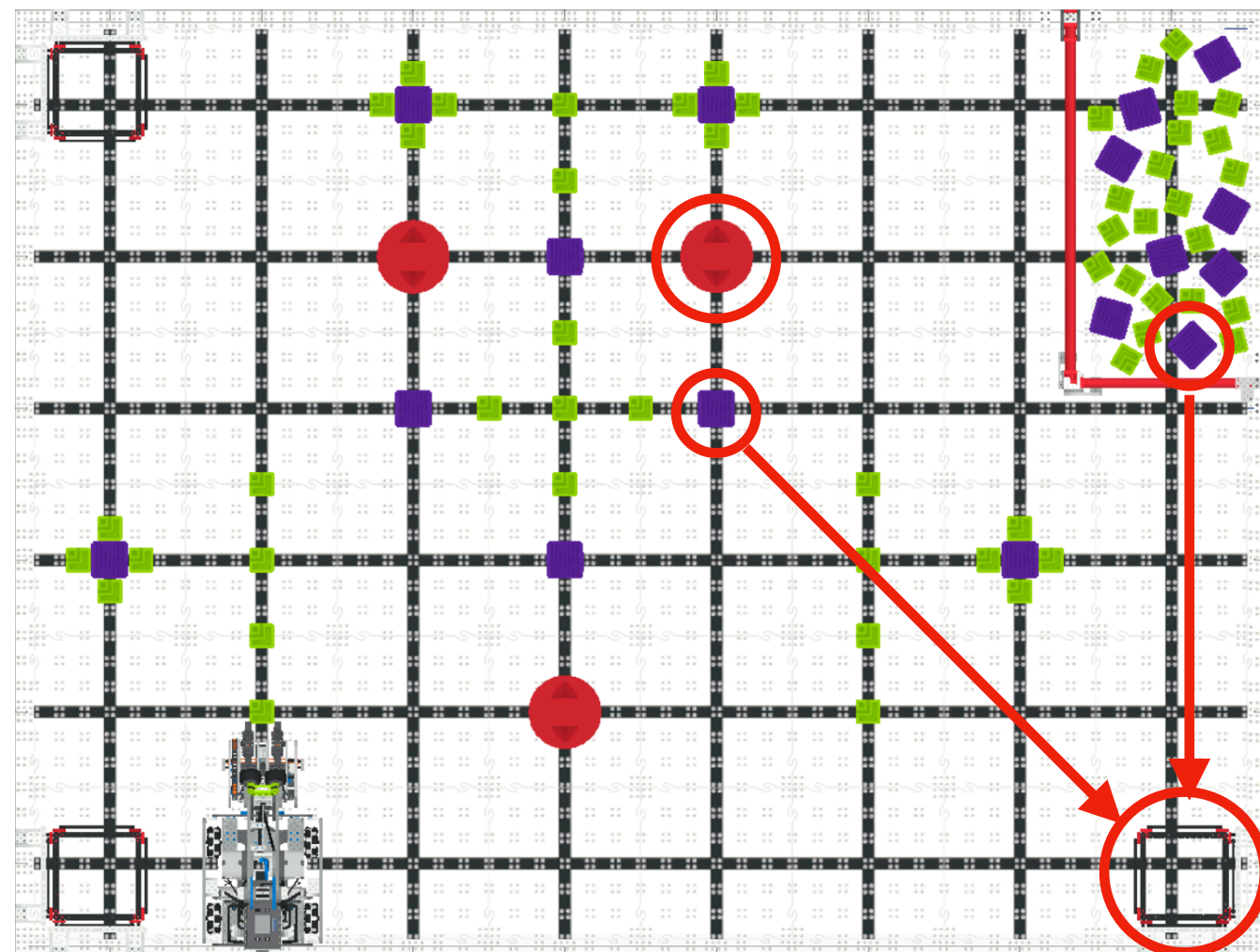
```

define Goal1Purple1
  intake the purple block
  drive reverse for 36 inches
  turn left for 90 degrees
  drive reverse for 26 inches
  turn left for 180 degrees
  rollArm toPos armIntake wait true
  intakeBlock
  drop the purple block
  rollArm toPos armIntakeRed wait false
  drive forward for 13.5 inches
  turn right for 105 degrees
  drive forward for 37.4 inches
  turn left for 105 degrees
  rollArm toPos armReverseDrop wait false
  drive reverse for 12.02 inches
  waitUtilArmComplete
  dropBlock
  
```

```

define Goal1Purple2
  intake the purple block
  rollArm toPos armIntake wait false
  drive forward for 12 inches
  waitUtilArmComplete
  drive forward for 1.6 inches
  intakeBlock
  drop the purple block
  rollArm toPos armReverseDrop wait false
  drive reverse for 13.6 inches
  waitUtilArmComplete
  wait 0.2 seconds
  dropBlock
  
```

Collect 2 purple blocks to Goal 1 to achieve Uniform and Fill Level 2 And a Red Block Removed from Starting Peg



To simulate the end state of the last step

Park

Using the **block enable/disable** function to enable debug mode

NOTE: Please follow the comments to set the startup location correctly

rolling the arm at **high speed** to achieve **fully parking**

```
define Park
  DEBUG start point: E
  E2Goal1PurpleDrop
  to park fully
  rollArm toPos armAvoidCube wait true
  turn right for 180 degrees
  to shake down Byte by rolling its arm
  rollArm toPos 750 wait false
  drive reverse for 11.9 inches
  waitUtilArmComplete
  rollArm toPos 1600 wait true
  rollArm toPos 0 wait true
```

```
define E2Goal1PurpleDrop
  rollArm toPos armAvoidCube wait false
  drive forward for 15.5 inches
  turn right for 90 degrees
  drive forward for 12 inches
  turn left for 90 degrees
  rollArm toPos armReverseDrop wait true
```

To **simulate** the end state of the last step

Advanced Utils

Advanced Arm Control

To support both sync/asyn arm rolling
using **conditional branches**

Using *armToPosition* variable
to **cache** the expected arm degrees

Using a **wait loop**
to wait for the robot arm
to reach the **cached** degrees
before the next step

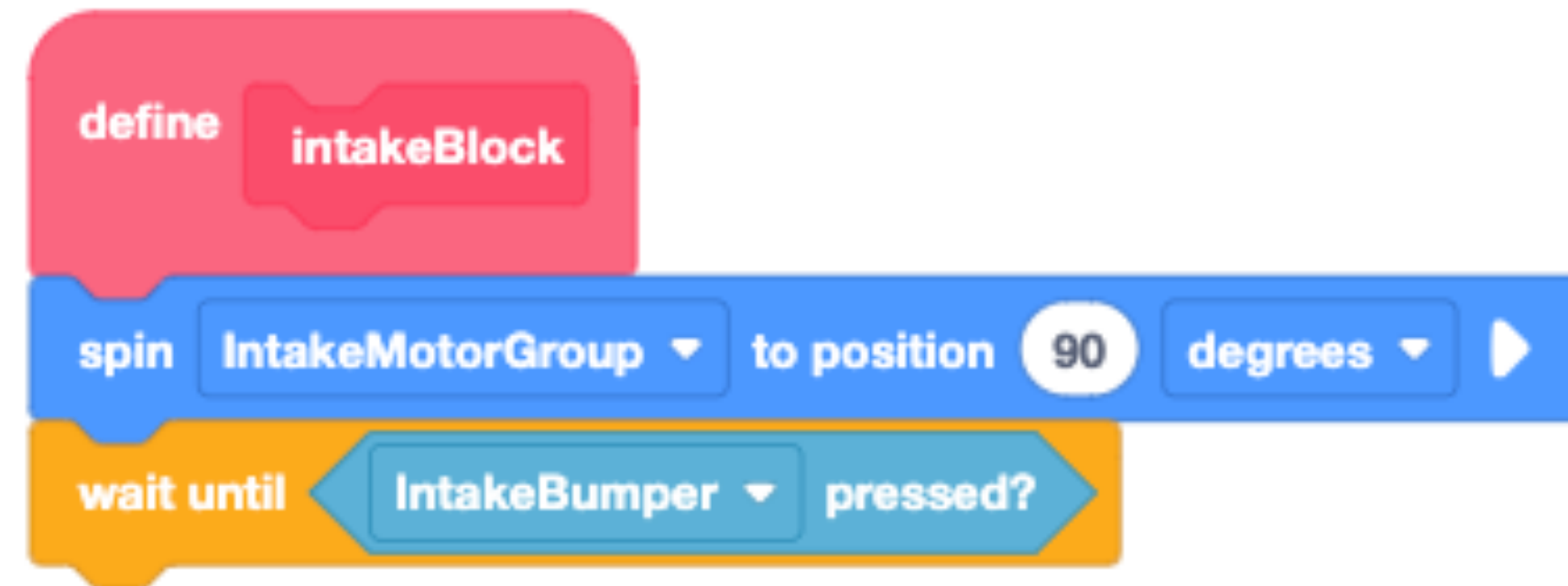
```
define rollArm toPos pos wait wait
if wait then
  spin ArmMotorGroup to position pos degrees
else
  spin ArmMotorGroup to position pos degrees and don't wait
  set armToPosition to pos

define waitUtilArmComplete
wait until ArmMotorGroup position in degrees = armToPosition
```

The image shows two Scratch code snippets. The first snippet defines a function 'rollArm toPos pos wait wait'. It contains an 'if wait then' conditional branch. The 'then' branch has a 'spin ArmMotorGroup to position pos degrees' block. The 'else' branch has a 'spin ArmMotorGroup to position pos degrees and don't wait' block followed by a 'set armToPosition to pos' block. The second snippet defines a function 'waitUtilArmComplete' which contains a 'wait until' block with the condition 'ArmMotorGroup position in degrees = armToPosition'.

Advanced Intake Controls

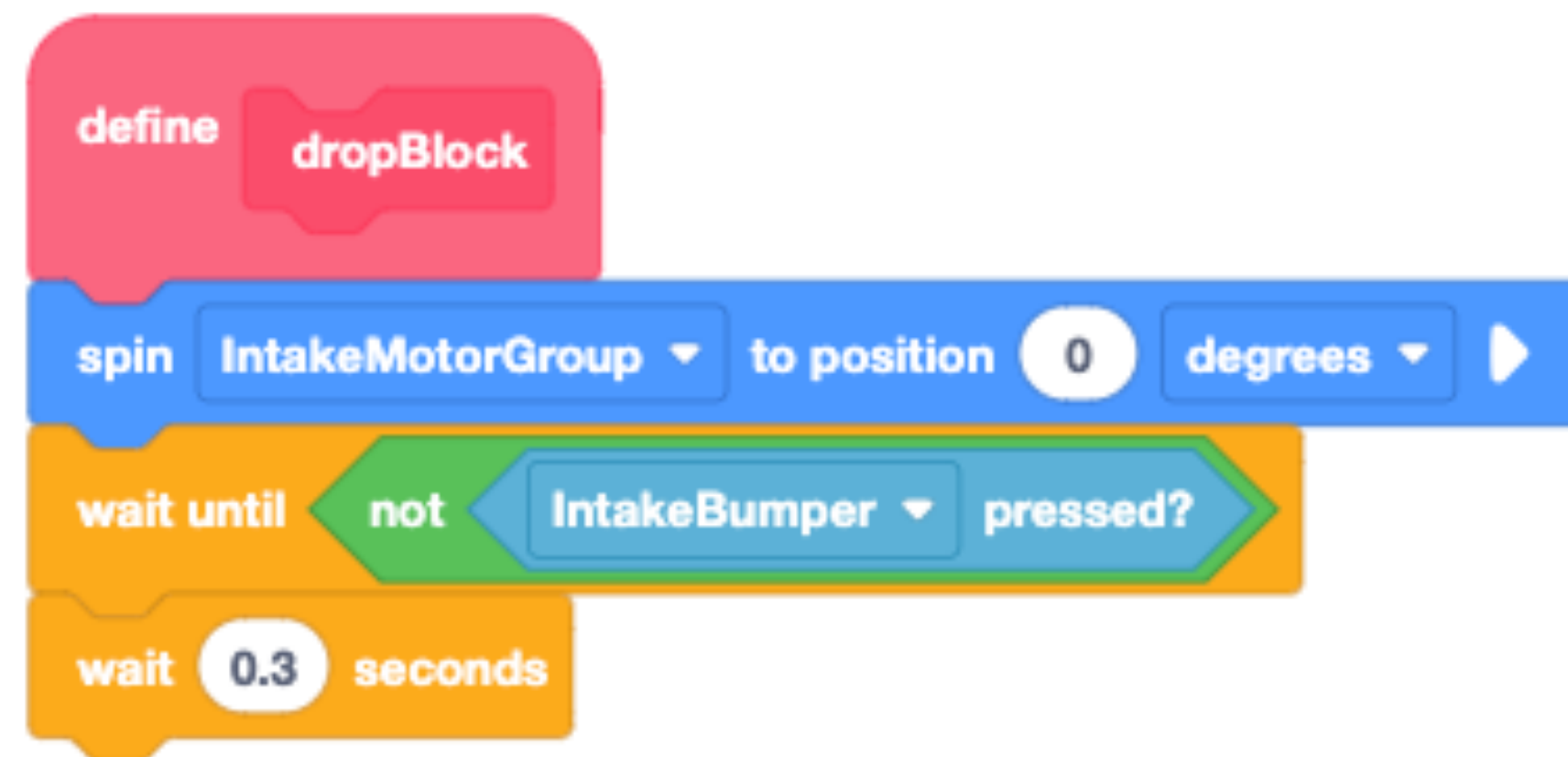
Enhanced **intake** function
using the **Bumper Switch sensor**
to ensure intaking completely



```
define intakeBlock
  spin IntakeMotorGroup to position 90 degrees
  wait until IntakeBumper pressed?
```

The code defines a function named 'intakeBlock'. It starts with a 'spin' block for 'IntakeMotorGroup' to rotate to 90 degrees. This is followed by a 'wait until' block that checks if the 'IntakeBumper' sensor is pressed.

Enhanced **drop** function
using **Bumper Switch sensor** and **timer**
to ensure dropping completely



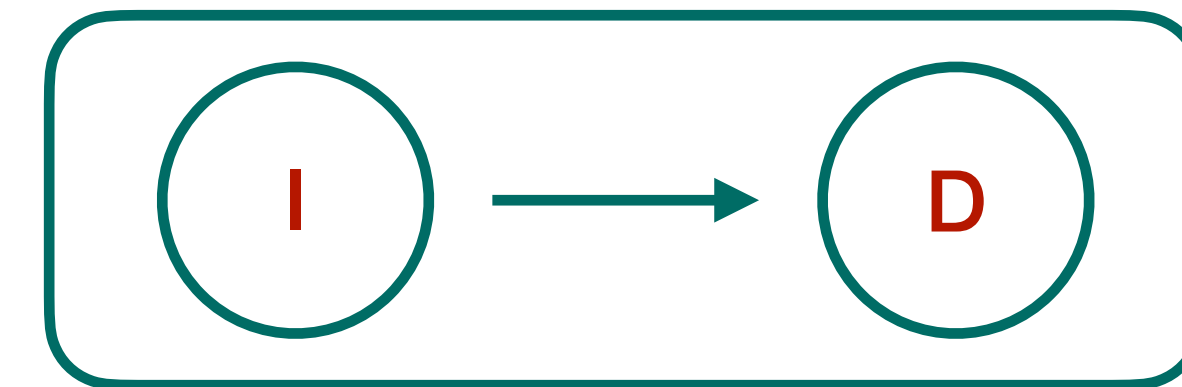
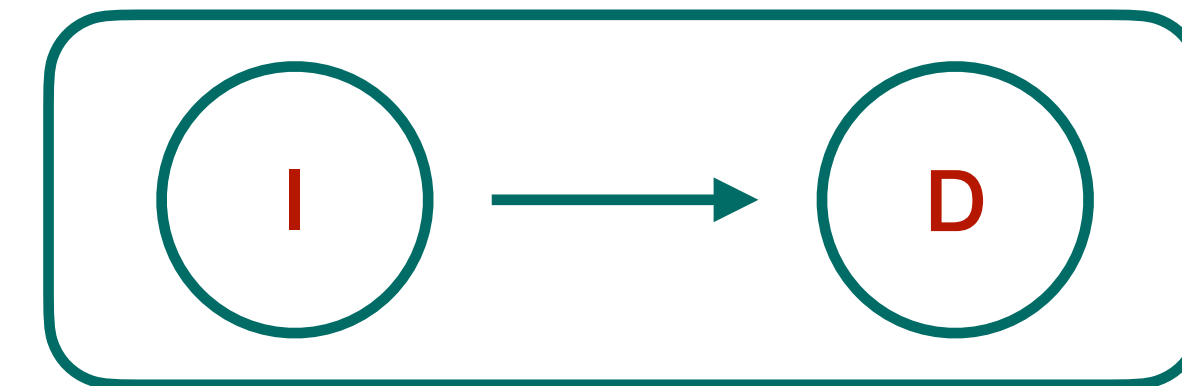
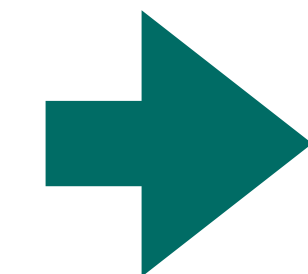
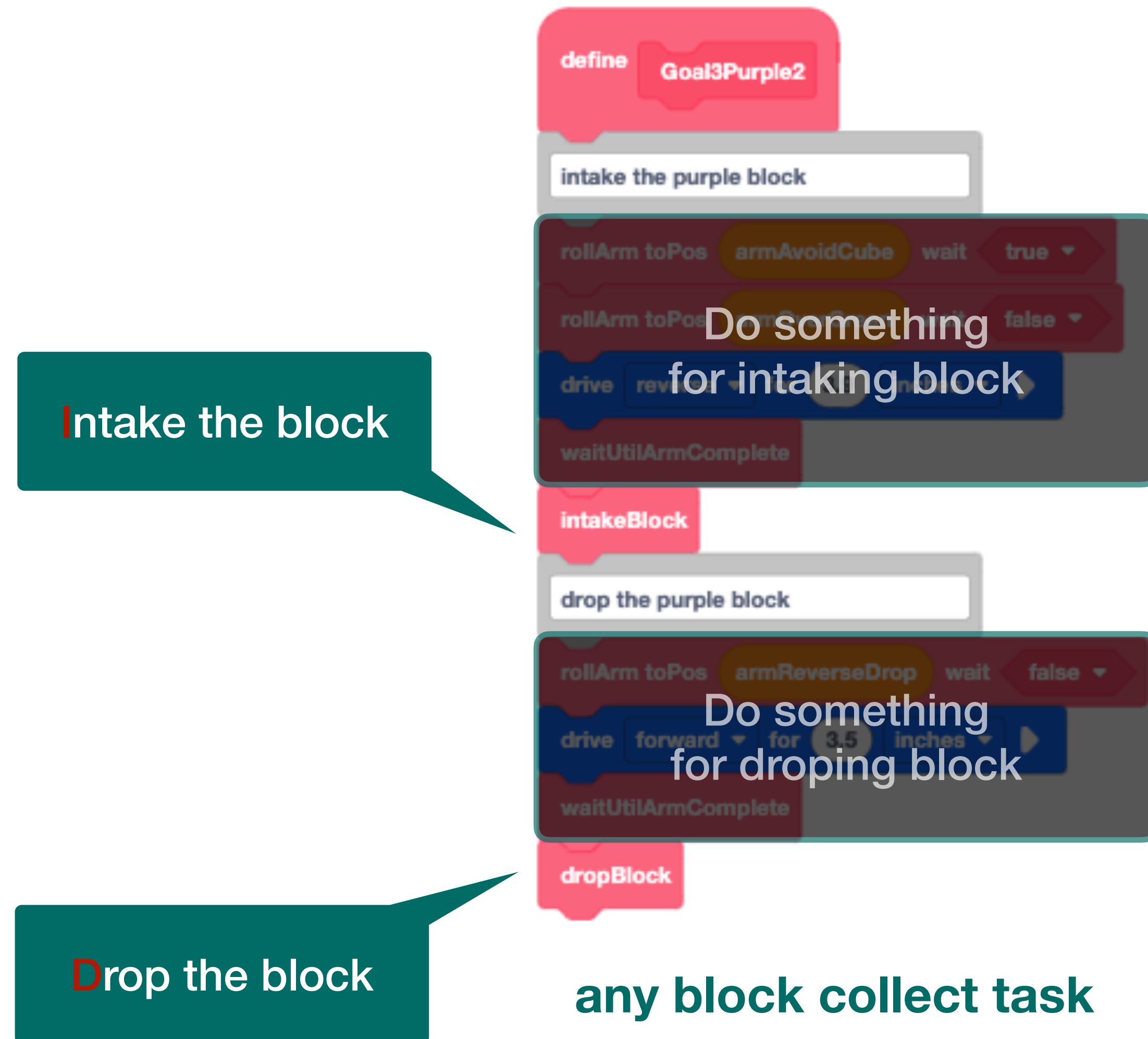
```
define dropBlock
  spin IntakeMotorGroup to position 0 degrees
  wait until not IntakeBumper pressed?
  wait 0.3 seconds
```

The code defines a function named 'dropBlock'. It starts with a 'spin' block for 'IntakeMotorGroup' to rotate to 0 degrees. This is followed by a 'wait until' block that checks if the 'IntakeBumper' sensor is **not** pressed. Finally, there is a 'wait' block for 0.3 seconds.

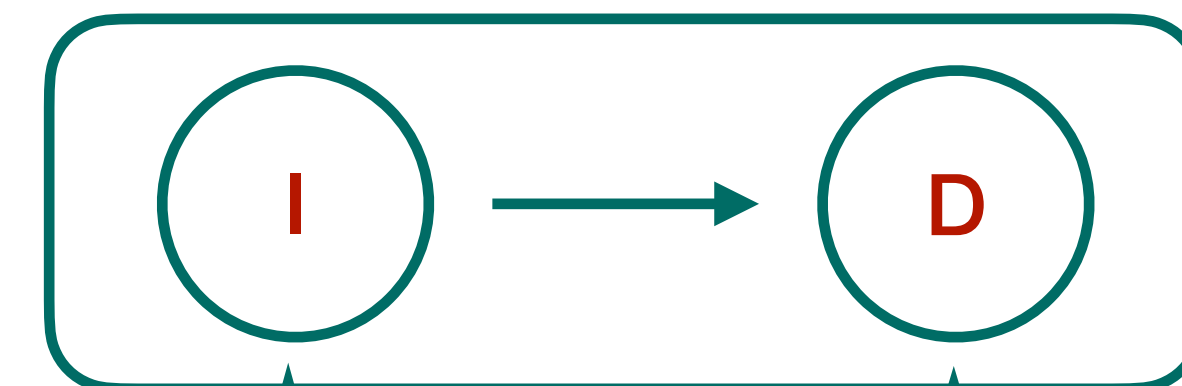
Intake Drop Pattern and Upgrade

We found in VR

Intake Drop Pattern



...

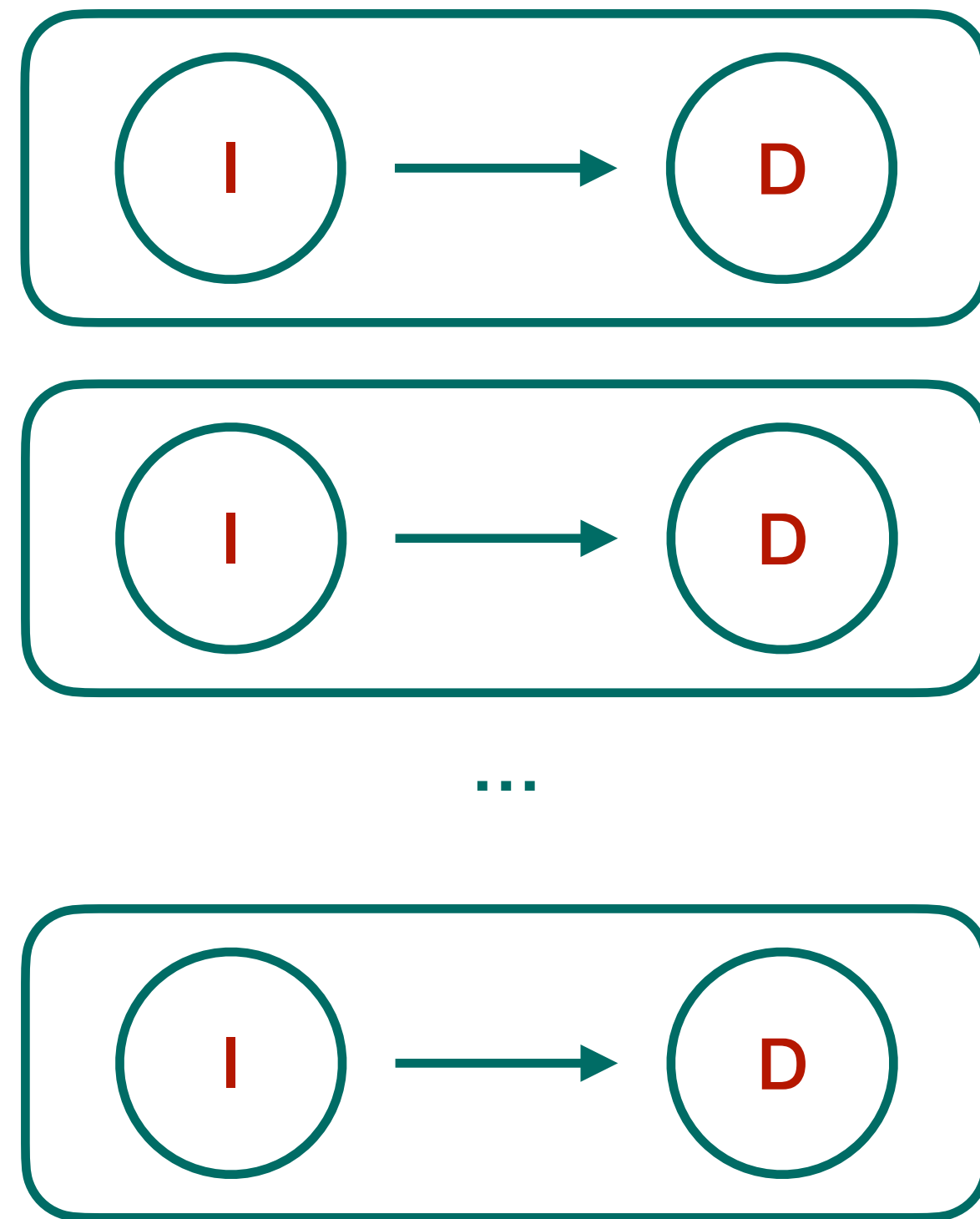


Intake 1 block per batch

Drop 1 block per batch

We Upgrade in VIQRC

Intake Drop Pattern



Collect **6** blocks in VR

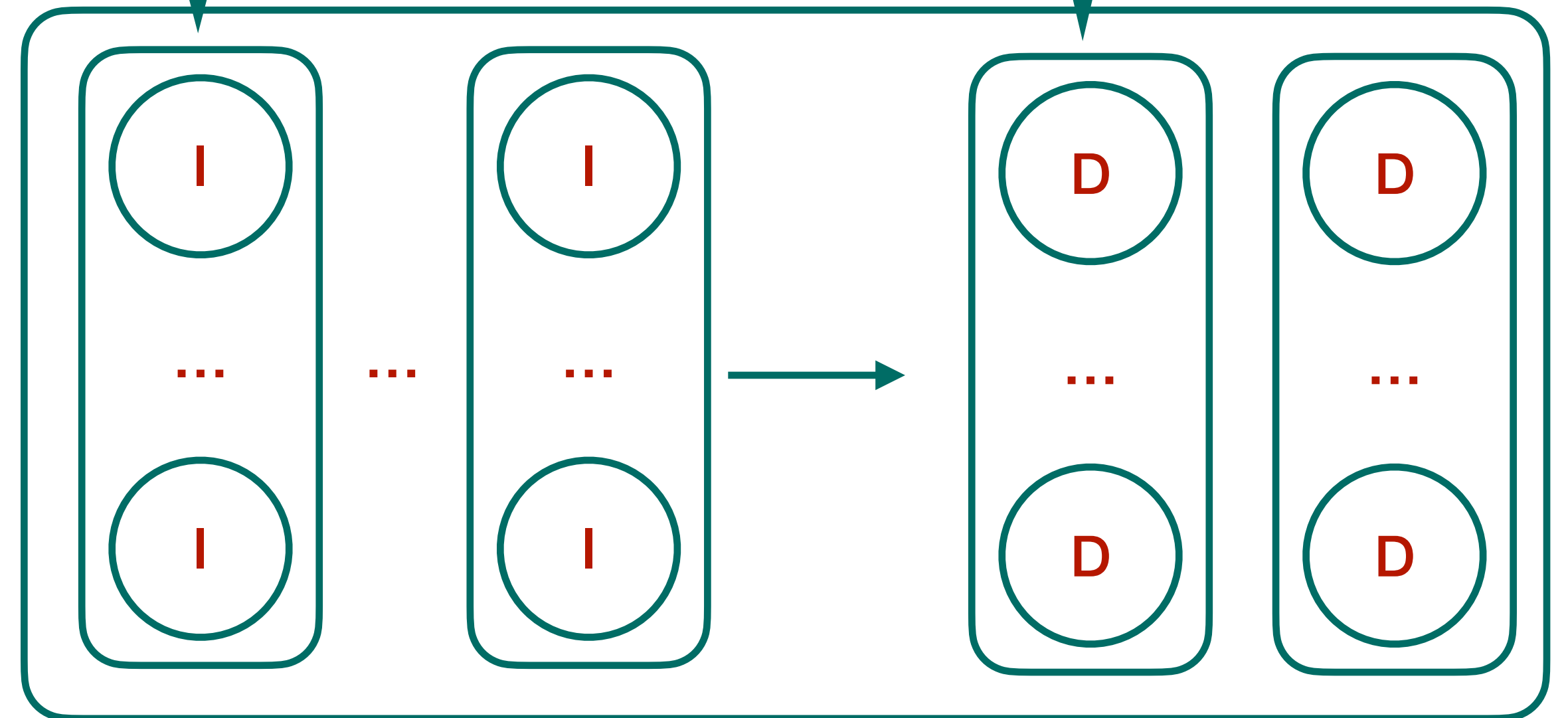
Upgrade
➔

We Need
Same drivetrain
Big intake
Big storage

Intake Drop ++

Intake n blocks per batch

Drop n blocks per batch



Collect **73** blocks in offline VIQRC

References

1. Getting Started with VIQRC '23-'24 Robot Design: Full Volume

<https://kb.vex.com/hc/en-us/articles/15957720967700-Getting-Started-with-VIQRC-23-24-Robot-Design-Full-Volume>

2. Get Started with VIQRC Full Volume Playground in VEXcode VR

<https://kb.vex.com/hc/en-us/articles/15571948163348-Get-Started-with-VIQRC-Full-Volume-Playground-in-VEXcode-VR>

3. Understanding Robot Features in VIQRC Full Volume for VEXcode VR

<https://kb.vex.com/hc/en-us/articles/15518491004692-Understanding-Robot-Features-in-VIQRC-Full-Volume-for-VEXcode-VR>

4. Understanding the VIQRC Full Volume Field Layout

<https://kb.vex.com/hc/en-us/articles/360060171872-Understanding-the-VIQRC-Full-Volume-Field-Layout>

5. VIQRC Full Volume | Hero Bot "Byte" | Part 1

<https://www.youtube.com/watch?v=E2-8dDP7Xrg>

6. VIQRC Full Volume | Hero Bot "Byte" | Part 2

<https://www.youtube.com/watch?v=AkOi6voPGgo>

7. Example: Remove Red Objects in VEXCode VR

8. Example: Scoring a Purple Object in VEXCode VR