Flex Marks - 3

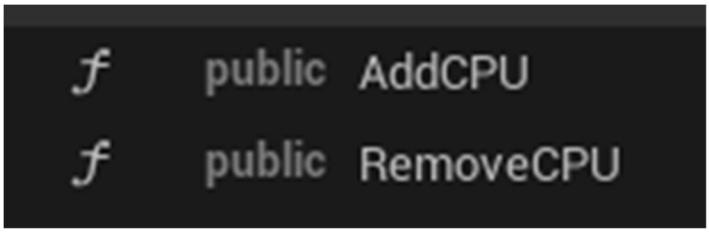
Added UI support for adding and removing CPU's when selecting a character

Why: Users need to be aware of how many CPU's have been added to the game. They also need to be able to add or remove CPU's at will to tailor the match to their preferences.

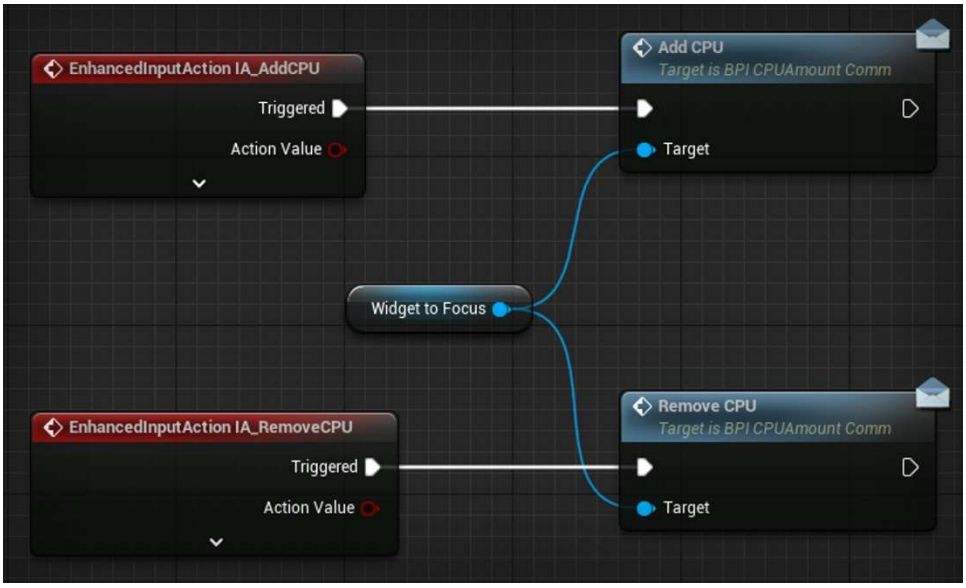
Adding and removing CPU's

Our team decision on how users could add/remove CPU's was to utilize the left and right triggers on the back of the controller. When the player is on the character select screen, by hitting the triggers you can toggle the amount of CPU's in your game.

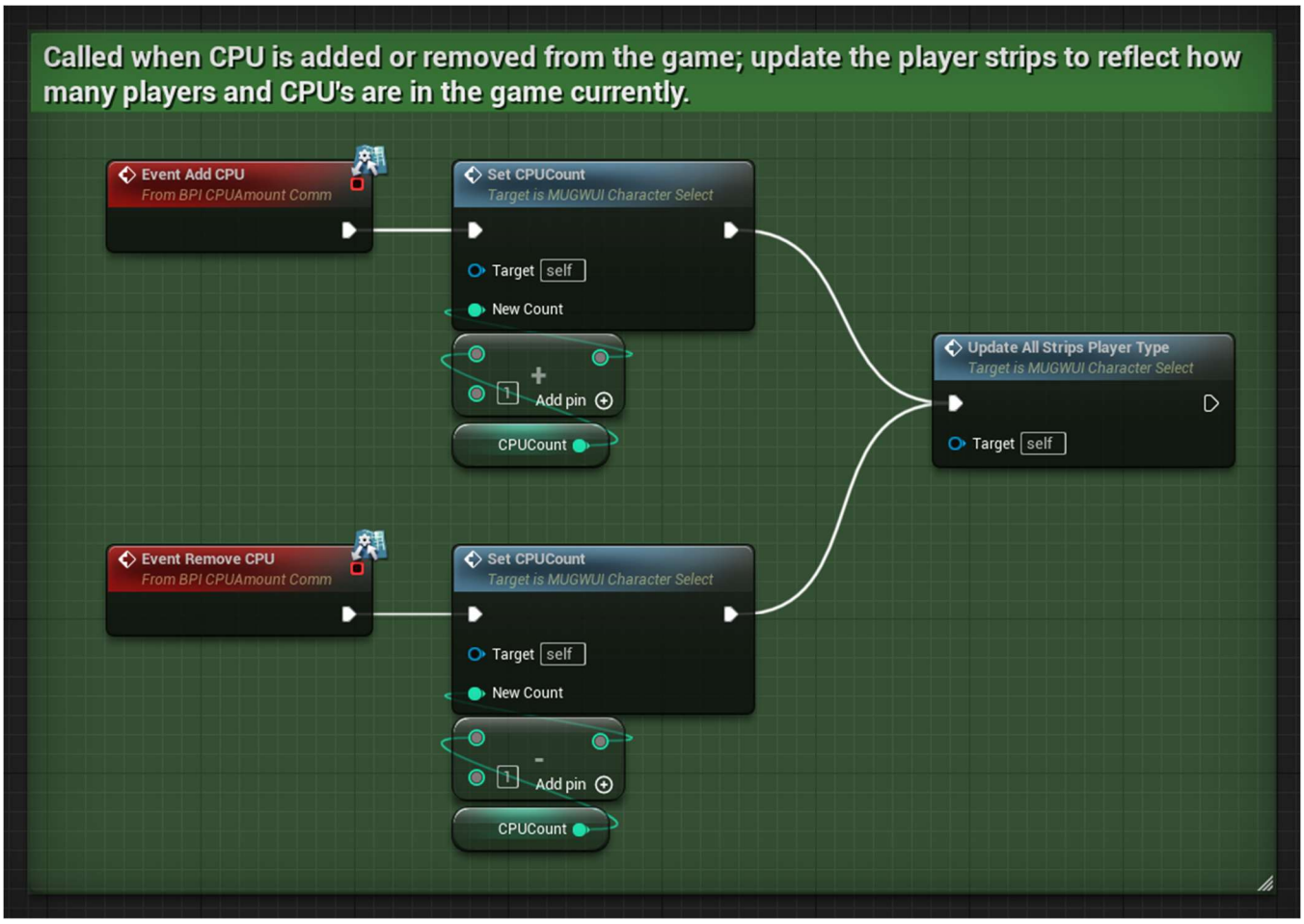
To communicate to the UI, I created a new Blueprint Interface integrated into `WBP_CharacterSelect` called `BPI_CPUAmountComm`. The `PC_Menu` controller then sends the `AddCPU`/`RemoveCPU` events to the `WBP_CharacterSelect` through the currently cached widget.



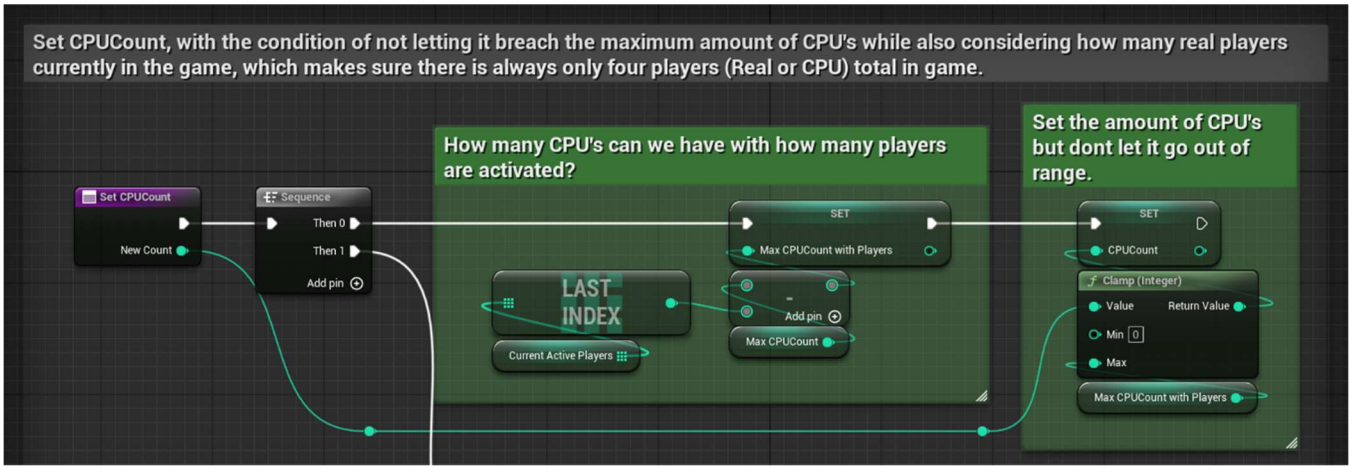
Functions available from `BPI_CPUAmountComm`



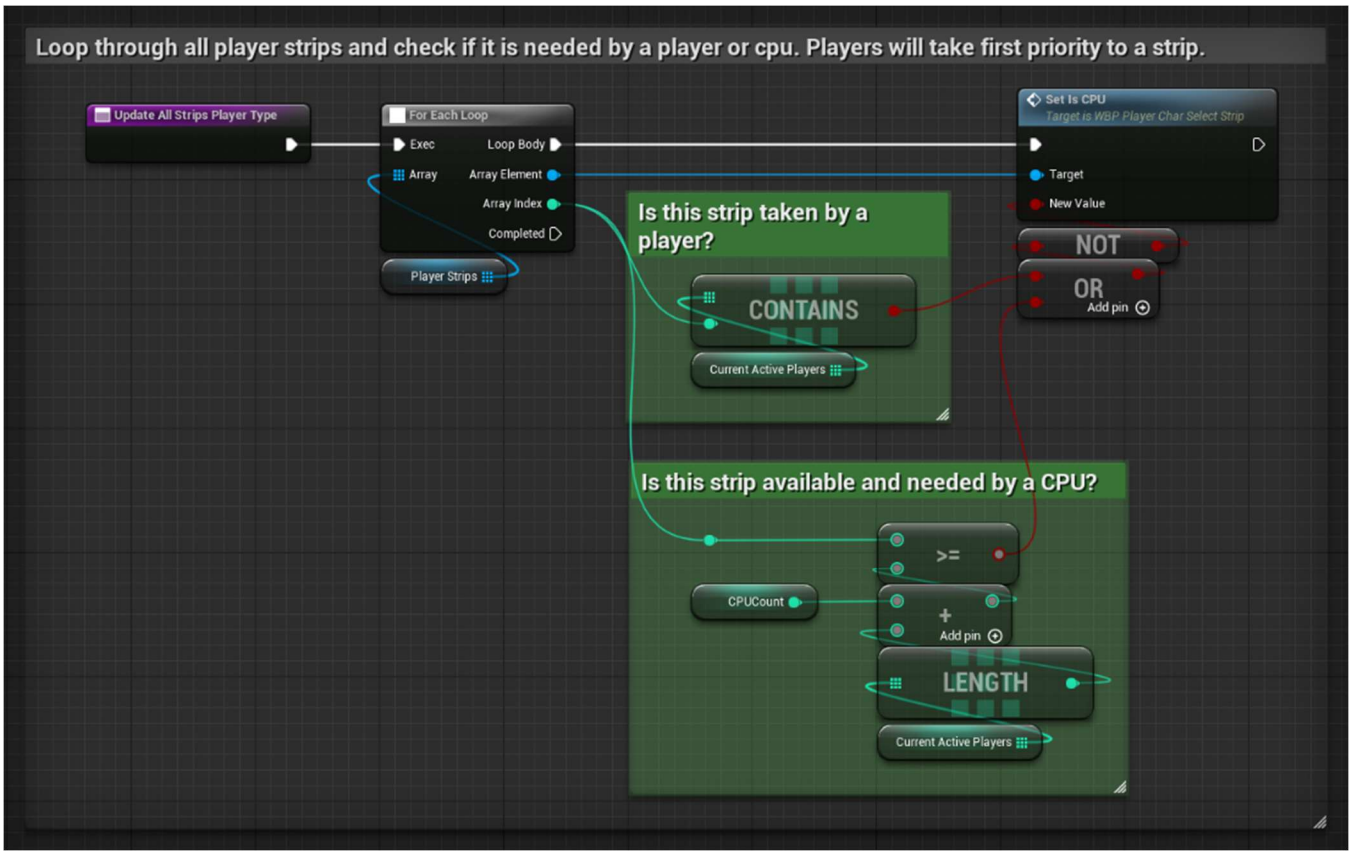
`PC_Menu`, calling `AddCPU` and `RemoveCPU` on the `WidgetToFocus` (which will only work on Character Select, since its the only one with the interface implemented.)



`WBP_CharacterSelect::AddCPU` & `WBP_CharacterSelect::RemoveCPU`, which will increment or decrement the `CPUCount` with `SetCPUCount`



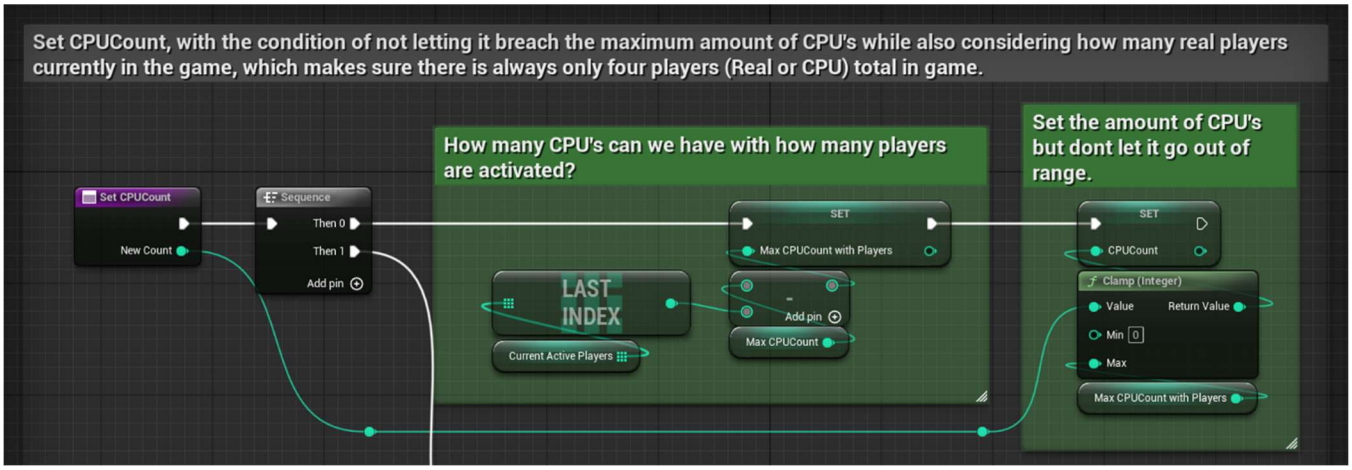
`WBP_CharacterSelect::SetCPUCount`, which will update the `CPUCount` while keeping it within the bounds of 4 players total. (The cut off part of this image is a debug print to notify how many CPU's there are, how many CPU's there can be, and if there are more CPU's than that amount).



`WBP_CharacterSelect::UpdateAllStripsPlayerType`, which will tell the Character Select Strips if they are for a CPU or not.

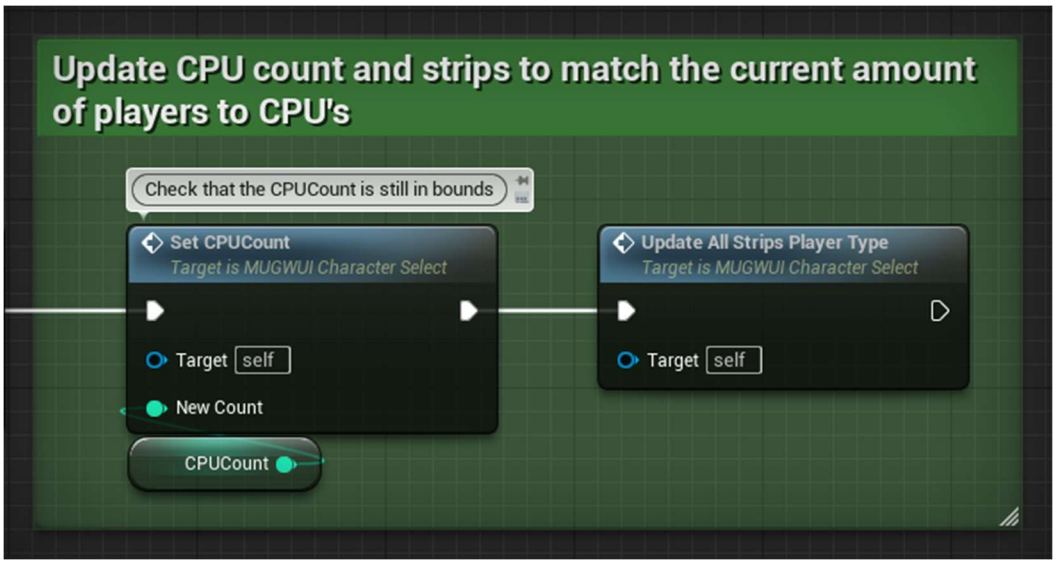
Player activation while CPU's are present

Because of the player limit (4 players max), there can only be as many CPU's as there are **inactivated** players. Every time one of the `BPI_CPUAmountComm` events are called in Character select, and since `CurrentActivePlayers` will have the correct amount of activated players, that can be used to calculate how many open slots are available for a CPU to fill.

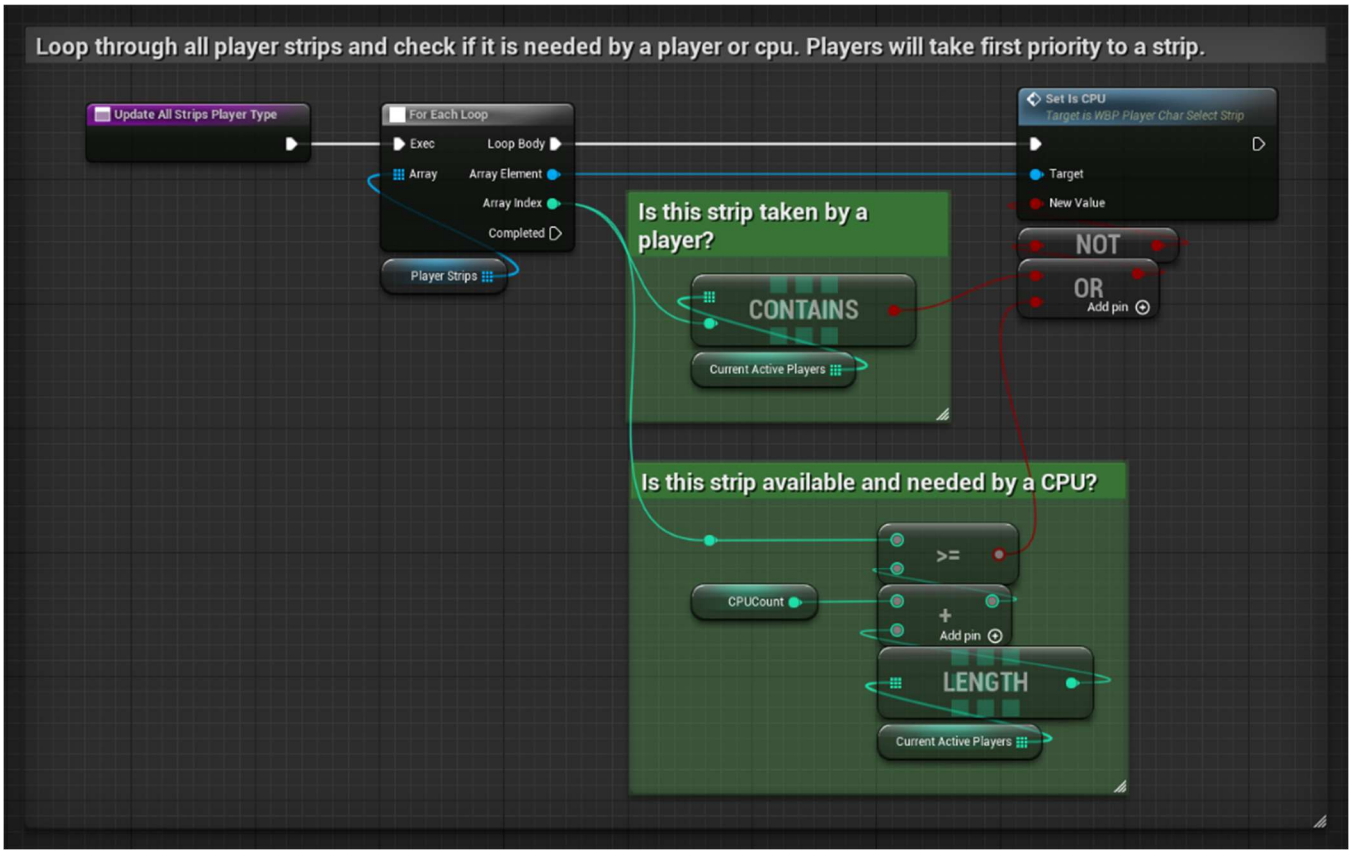


`WBP_CharacterSelect::SetCPUCount`, which will update the `CPUCount` while keeping it within the bounds of 4 players total. (The cut off part of this image is a debug print to notify how many CPU's there are, how many CPU's there can be, and if there are more CPU's than that amount).

And then, when `WBP_CharacterSelect::PlayerActivated` is called, it will run `SetCPUCount` and `UpdateAllStripsPlayerType` to make sure that there isn't too many CPU's left and "insert" the new player between the CPUs and existing players.



`SetCPUCount` and `UpdateAllStripsPlayerType` at the tail end of `WBP_CharacterSelect::PlayerActivated`



`WBP_CharacterSelect::UpdateAllStripsPlayerType`, which will tell the Character Select Strips if they are for a CPU or not.