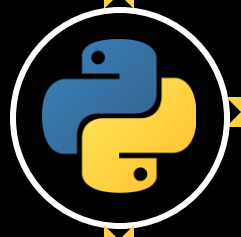
Introduction to Computer Programming

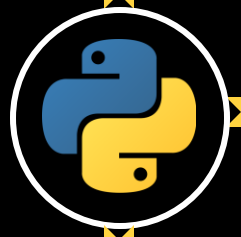
Murat Ugur KİRAZ
Online Python Training

How Does a Computer Work?



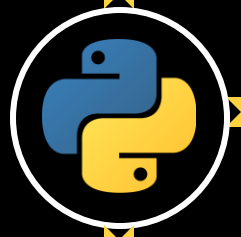
- The program makes the computer usable.
- Computers can perform very complex tasks.
- It can only perform extremely simple operations.
- Contemporary computers can only evaluate the results of very basic operations such as addition or division.

Natural languages and programming languages



- Computers also have a very primitive language called machine language.
- A computer, even the most technically complex, lacks even the slightest trace of intelligence.
- A complete set of known instructions is called an instruction list and sometimes abbreviated as IL.
- Currently, no computer is capable of creating a new language.

What is Required for a Language to Become a "Language"?



- Every language consists of the following elements:
  - Alphabet
  - Symbol
  - syntax
  - Semantics
- IL (Instruction List) is actually the alphabet of a machine language. It is the native language of the computer.
- Basic level languages
- High Level languages

History

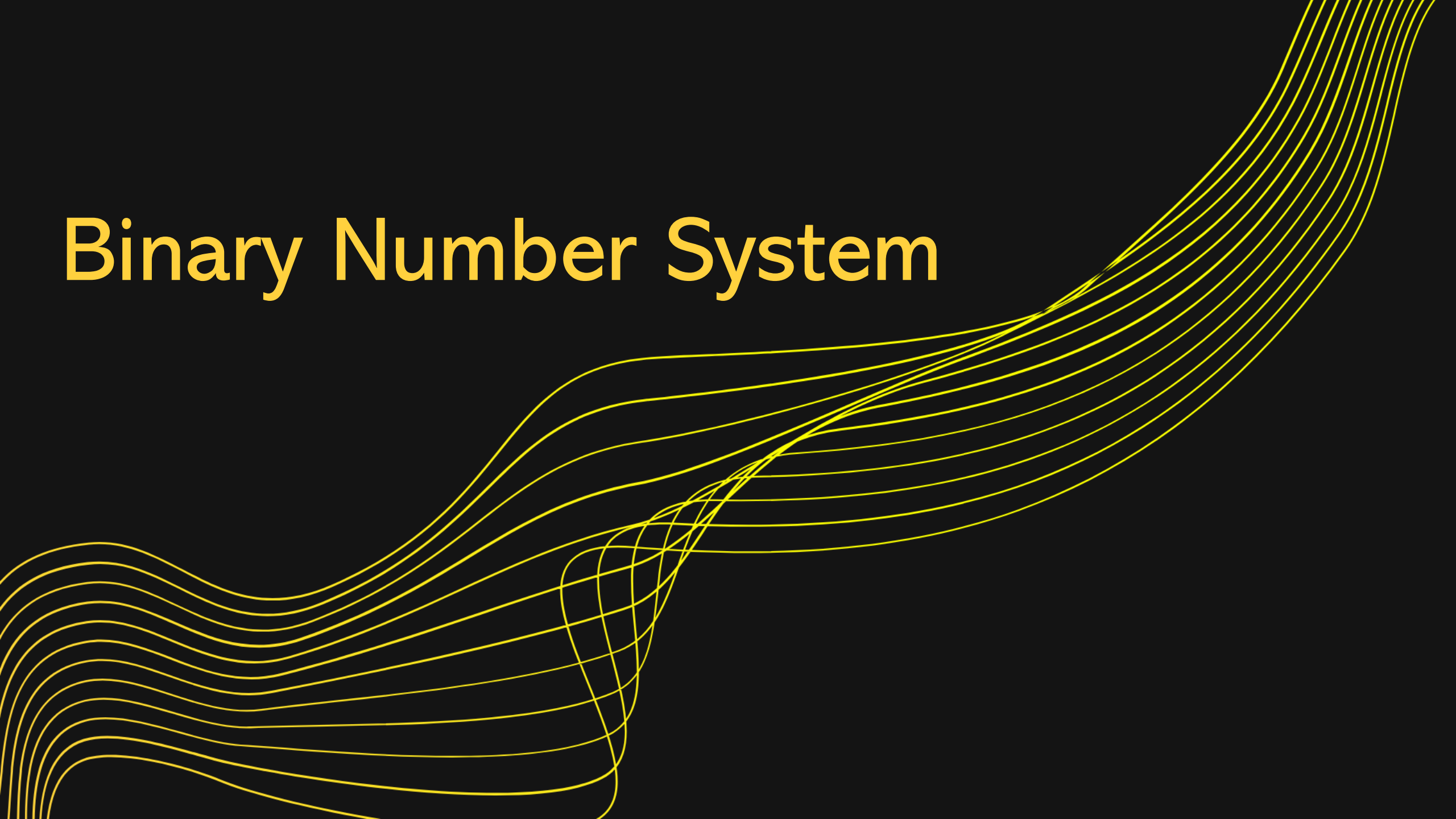
Birth of Technology

- Mechanical Era
- Electro-Mechanical Era
- Electronic Era

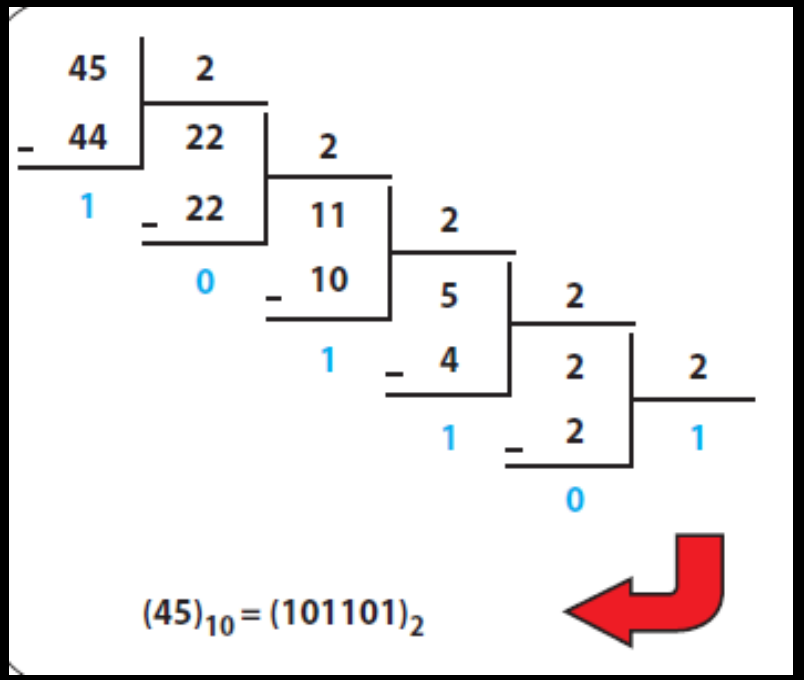
- First Generation Computers (1940 – 1956)
- Second Generation Computers (1956 – 1963)
- Third Generation Computers (1946 -1971)
- Fourth Generation Computers (1971 – Present)
- Fifth Generation Computers (Present – Near Future

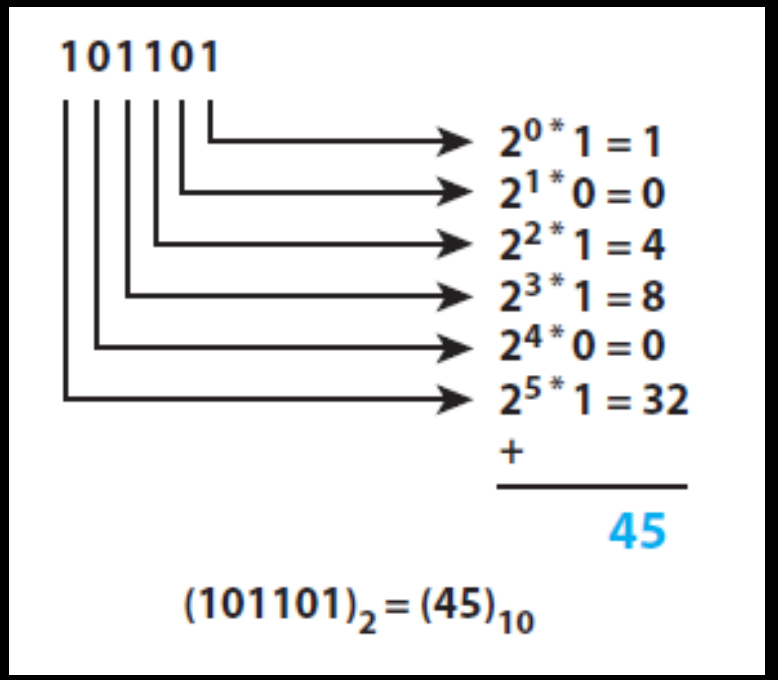
First Computer

Binary Number System

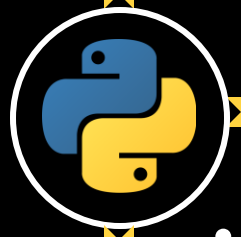


Binary, Number System





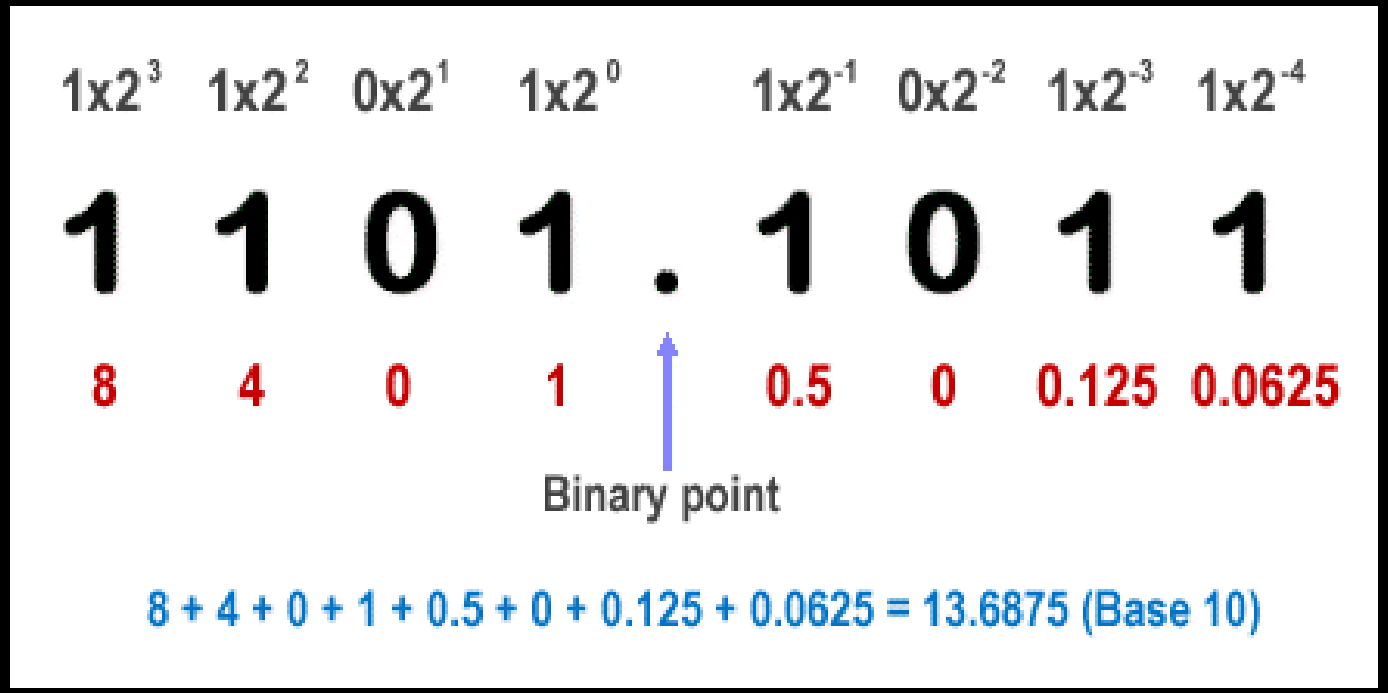
Binary Expression



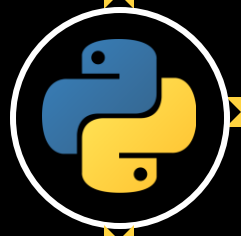
- numbers,
- texts,
- Colours, Photos, Video
- Sound,

- 8 Bit = 1 byte
- 1000 bytes = 1 Kilobyte
- 1000 Kilobytes = 1 Megabyte
- 1000 Megabytes = 1 Gigabyte
- 1000 Gigabytes = 1 Terrabyte

Fractional Numbers



Texts



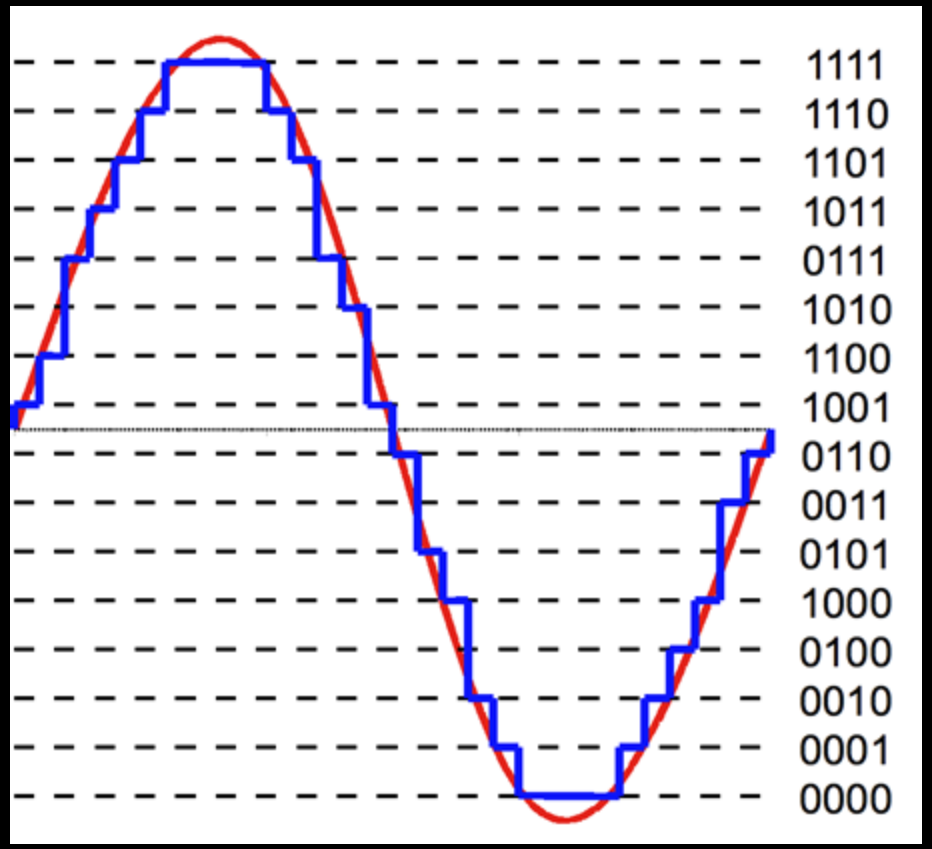
- ASCII table
- UTF Encoding

Colours, Photos, Video

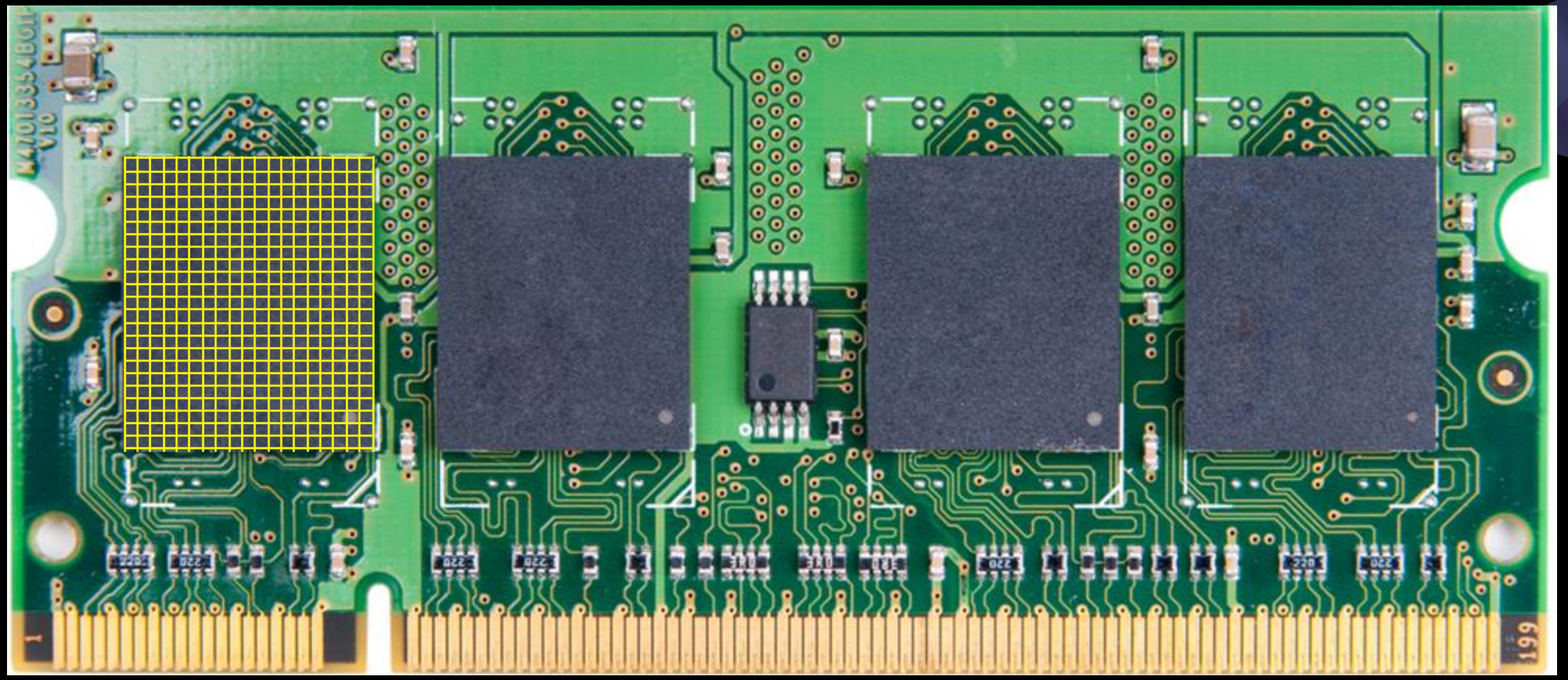
- RGB Value
- pixel

Sound

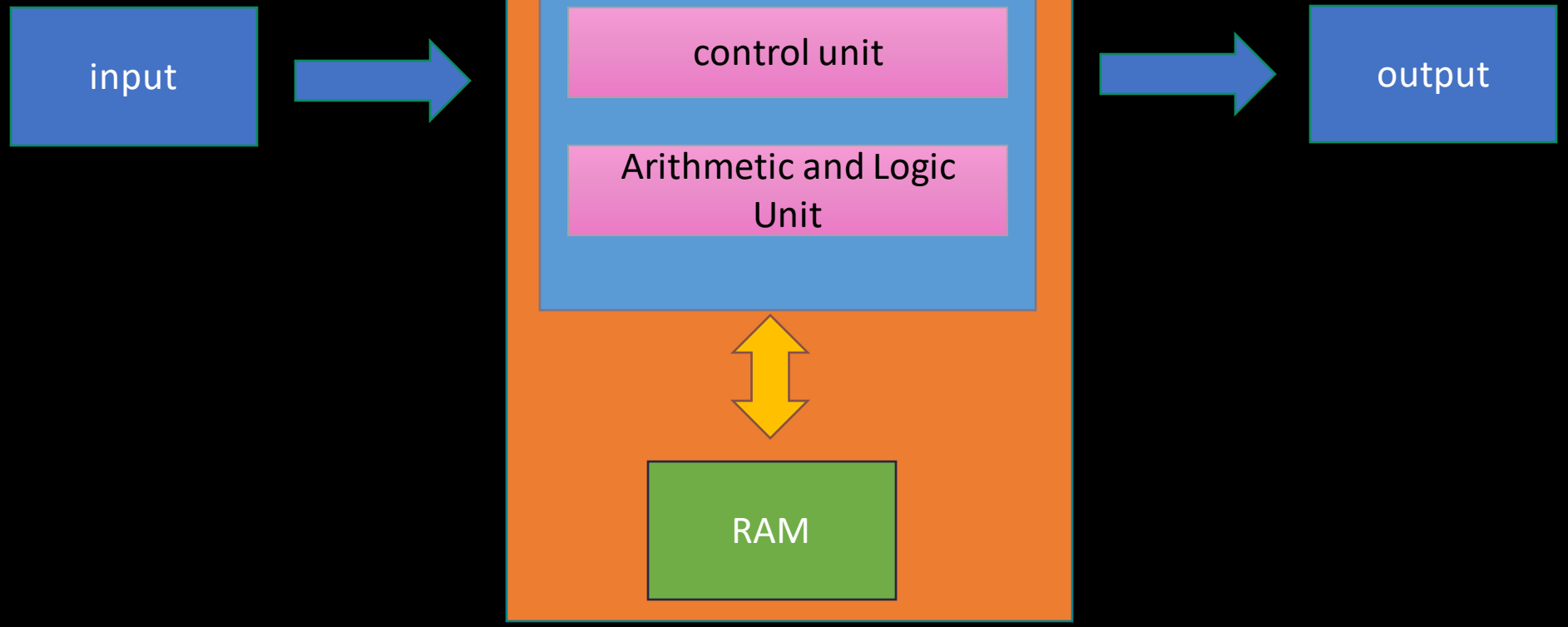
- Analog-Digital conversion



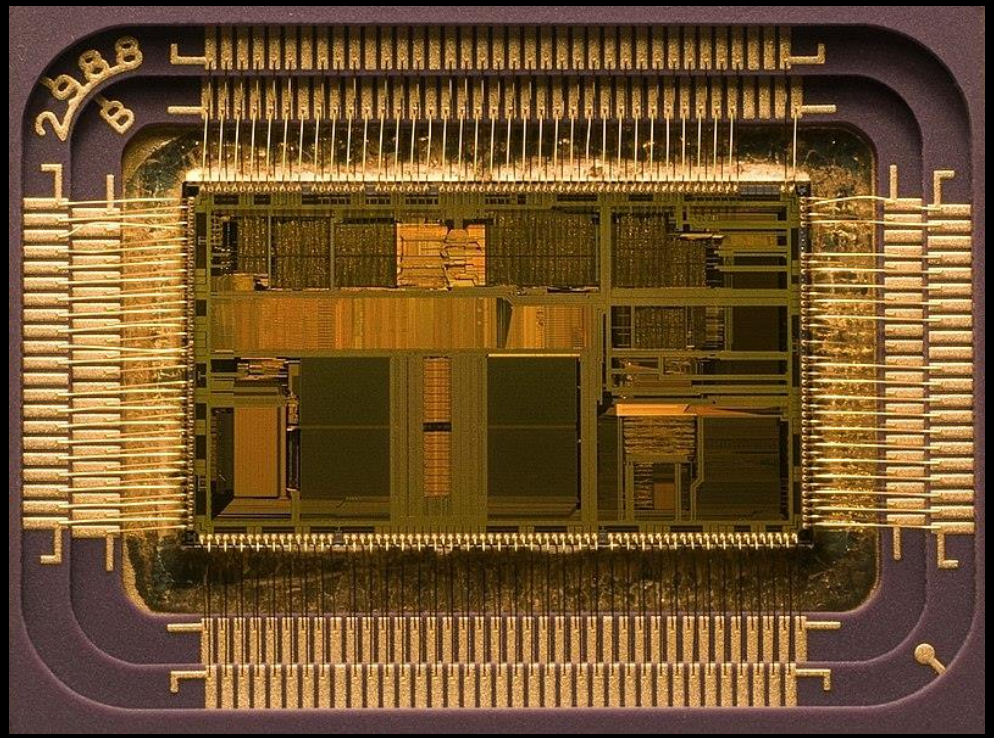
RAM



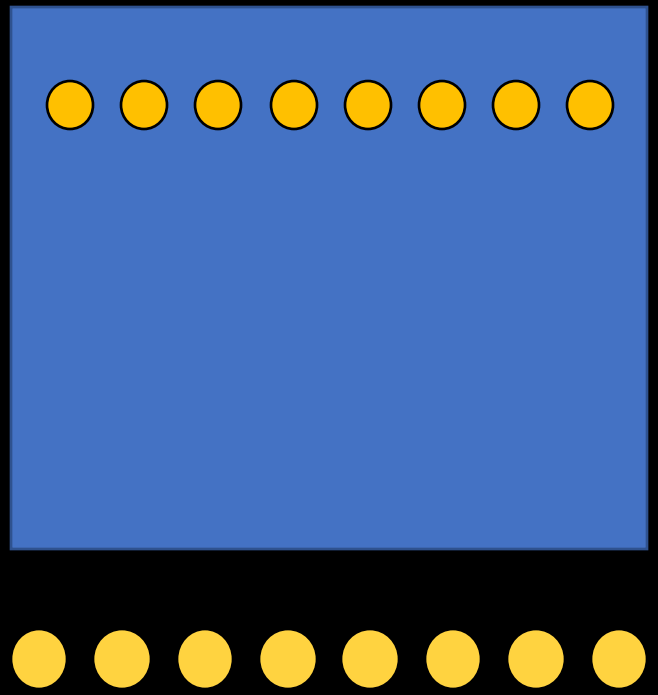
Computer Architecture



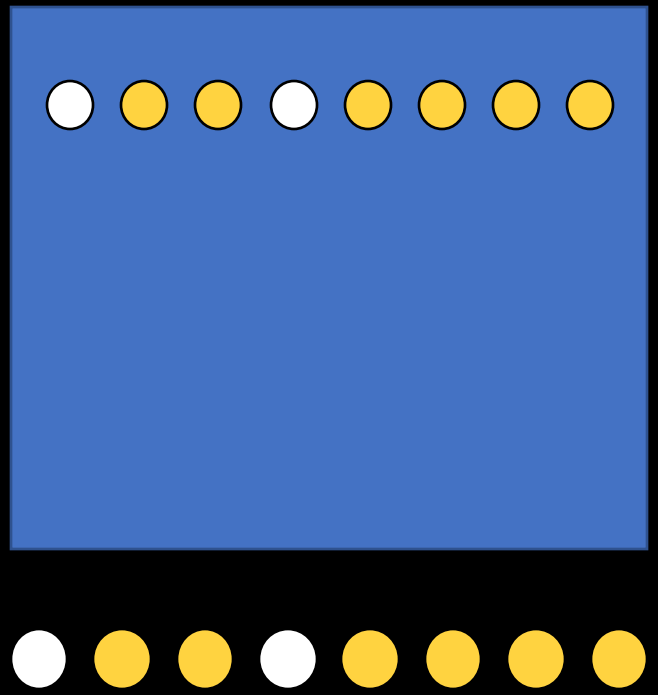
Central Processing Unit



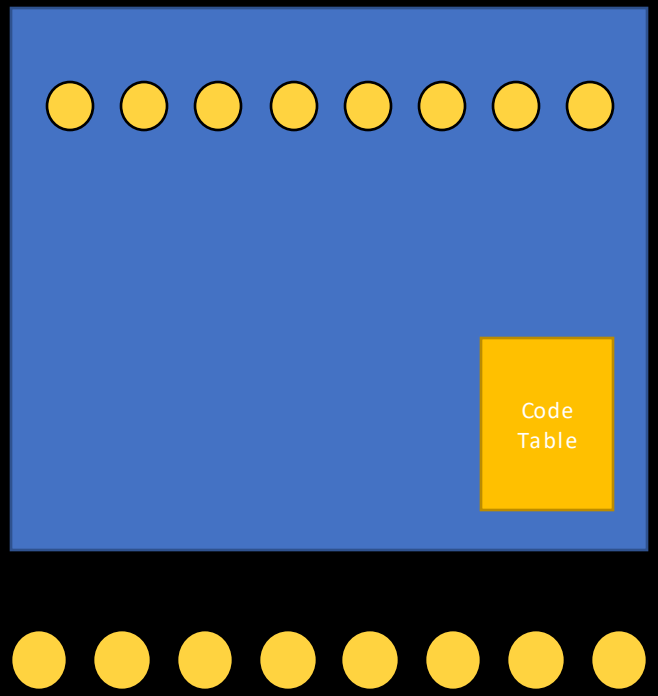
Central Processing Unit



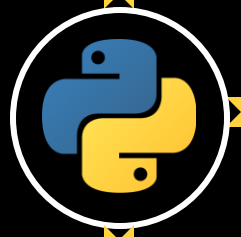
Central Processing Unit



Central Processing Unit

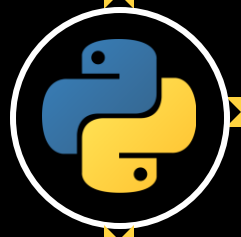


Central Processing Unit – Code Table



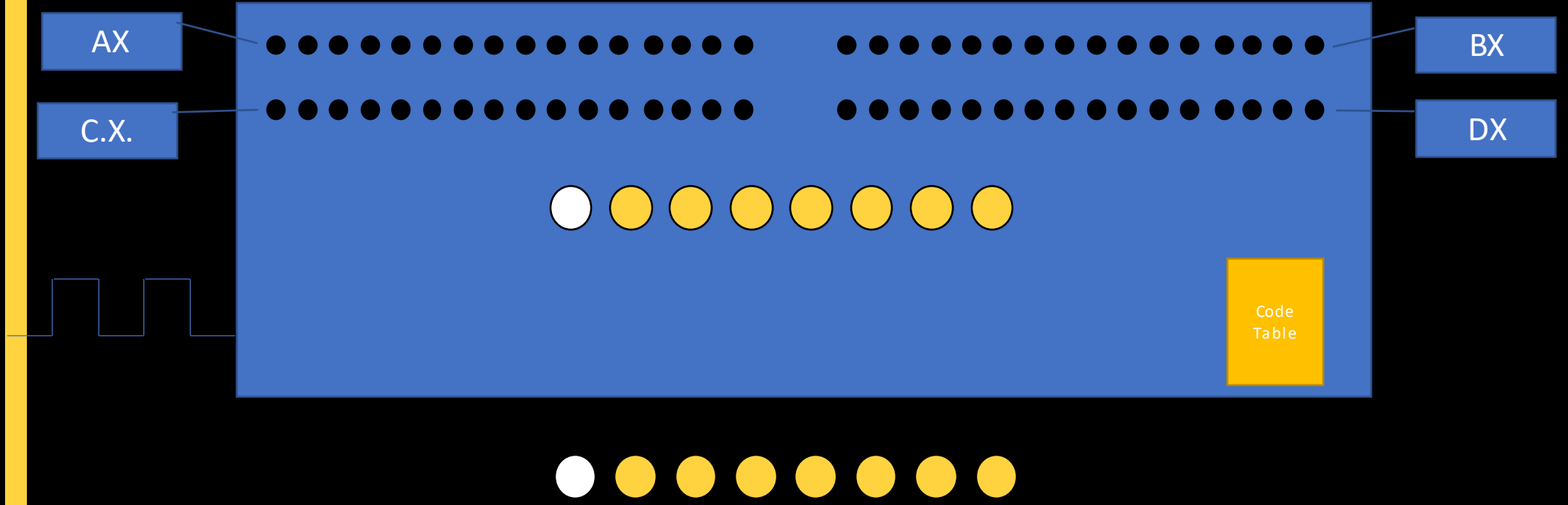
1000 0000	Next up is a number. Place it in the AX register.
1001 0000	Next up is a number. Place it in the BX register.
1011 0000	Add AX and BX, write the transaction to the AX register
1100 0000	Write the result to the external bus
0000 0000	0
0000 0001	1
0000 0010	2
0000 0011	3

A Simple Addition

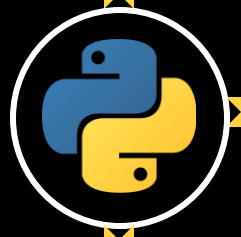


- 3+5 = ?
- 3 = 0000 0011
- 5 = 0000 0101
- 8 = 0000 1000

Central Processing Unit

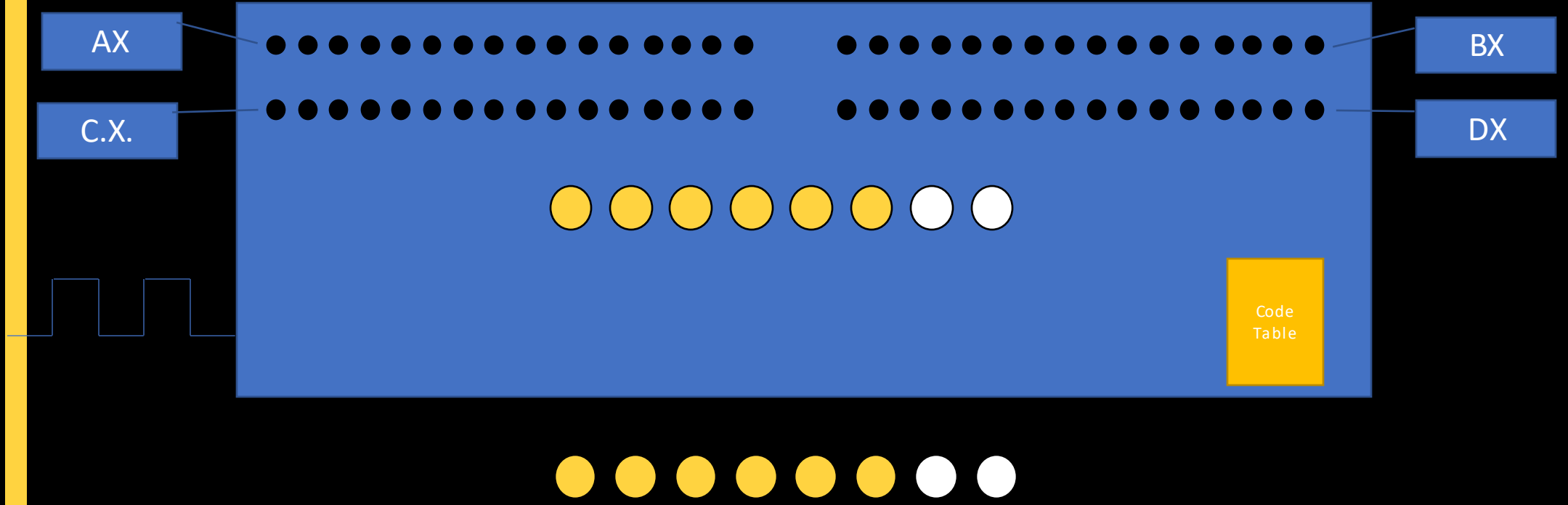


Central Processing Unit – Code Table

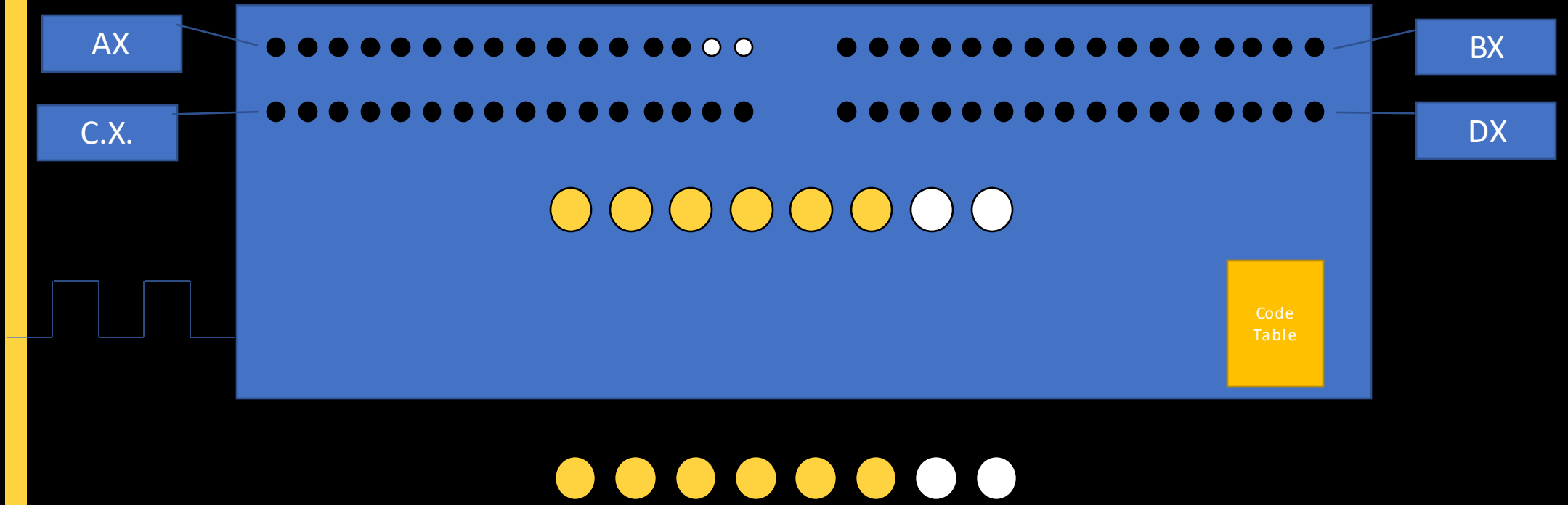


1000 0000	Next up is a number. Place it in the AX register.
1001 0000	Next up is a number. Place it in the BX register.
1011 0000	Add AX and BX, write the transaction to the AX register
1100 0000	Write the result to the external bus
0000 0000	0
0000 0001	1
0000 0010	2
0000 0011	3

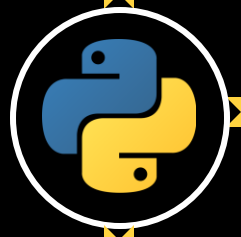
Central Processing Unit



Central Processing Unit

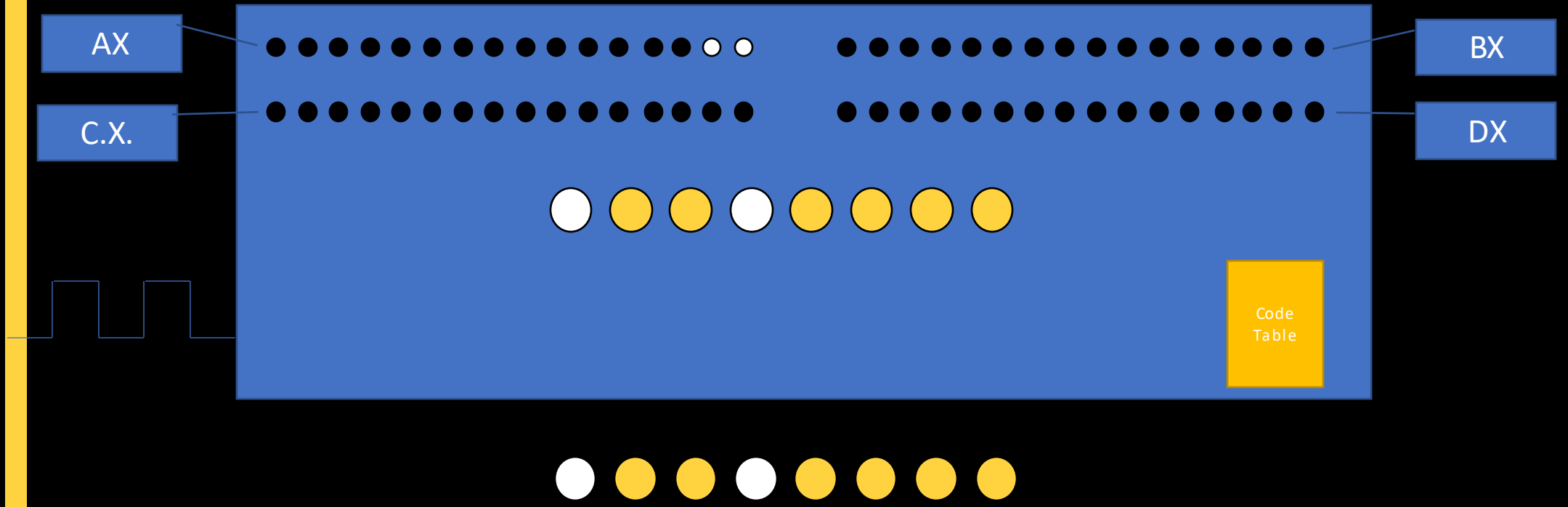


Central Processing Unit – Code Table

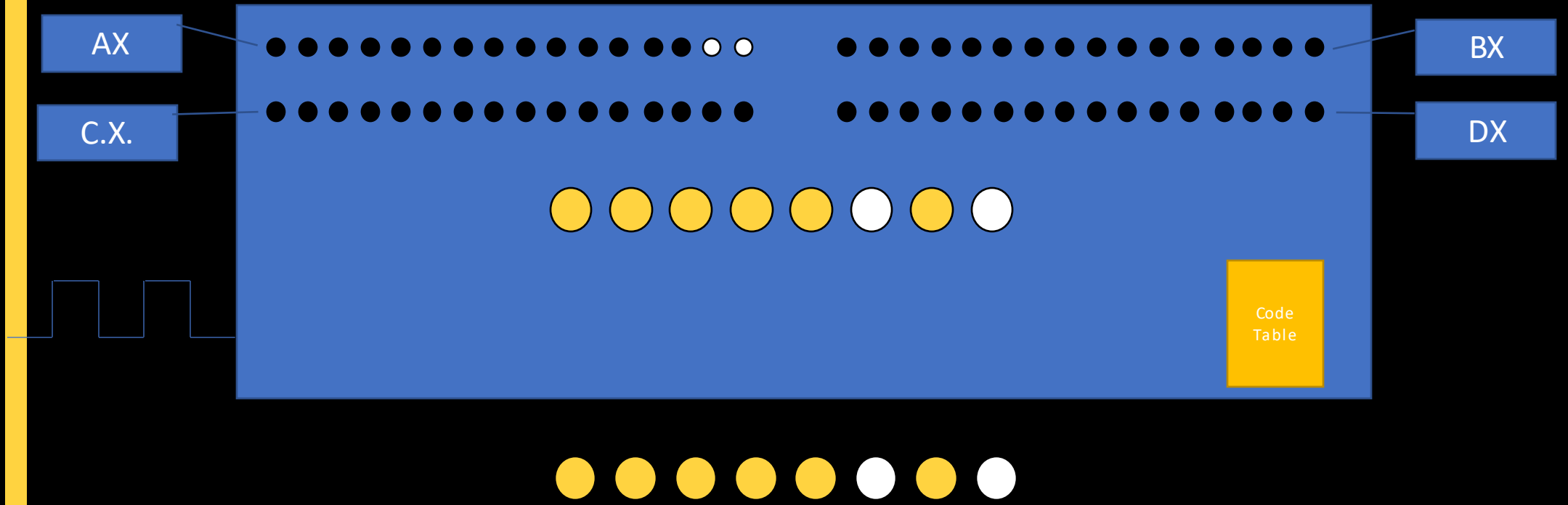


1000 0000	Next up is a number. Place it in the AX register.
1001 0000	Next up is a number. Place it in the BX register.
1011 0000	Add AX and BX, write the transaction to the AX register
1100 0000	Write the result to the external bus
0000 0000	0
0000 0001	1
0000 0010	2
0000 0011	3

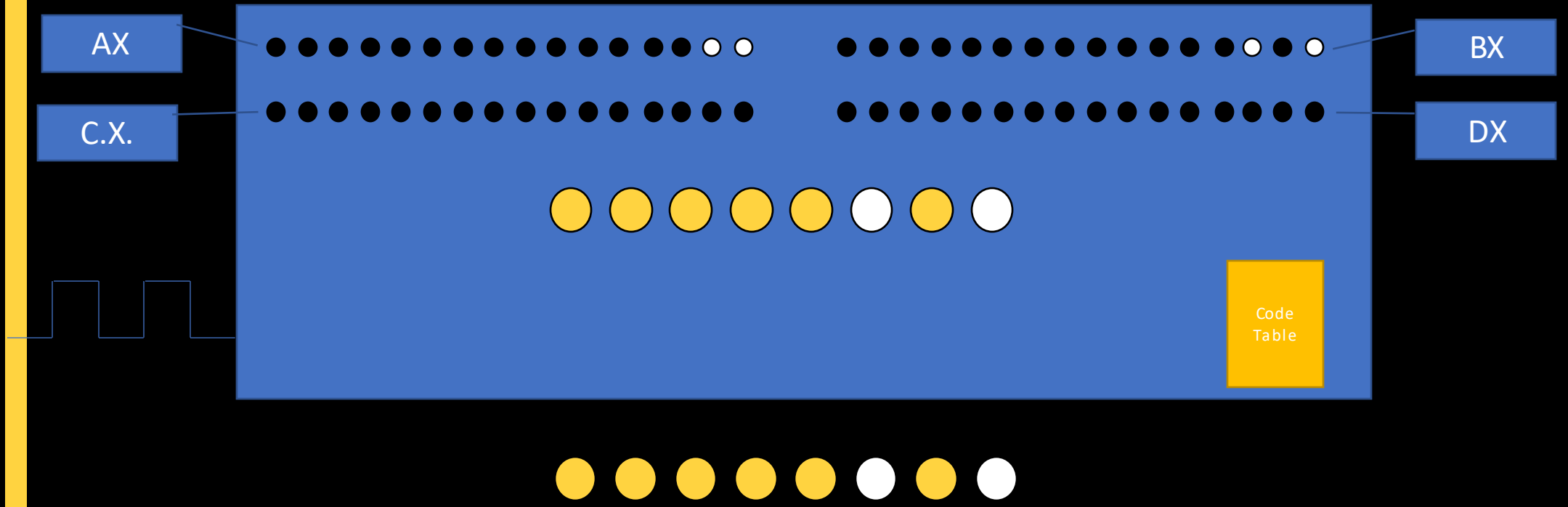
Central Processing Unit



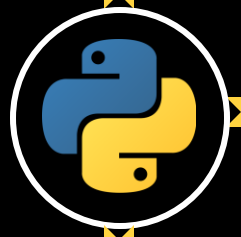
Central Processing Unit



Central Processing Unit

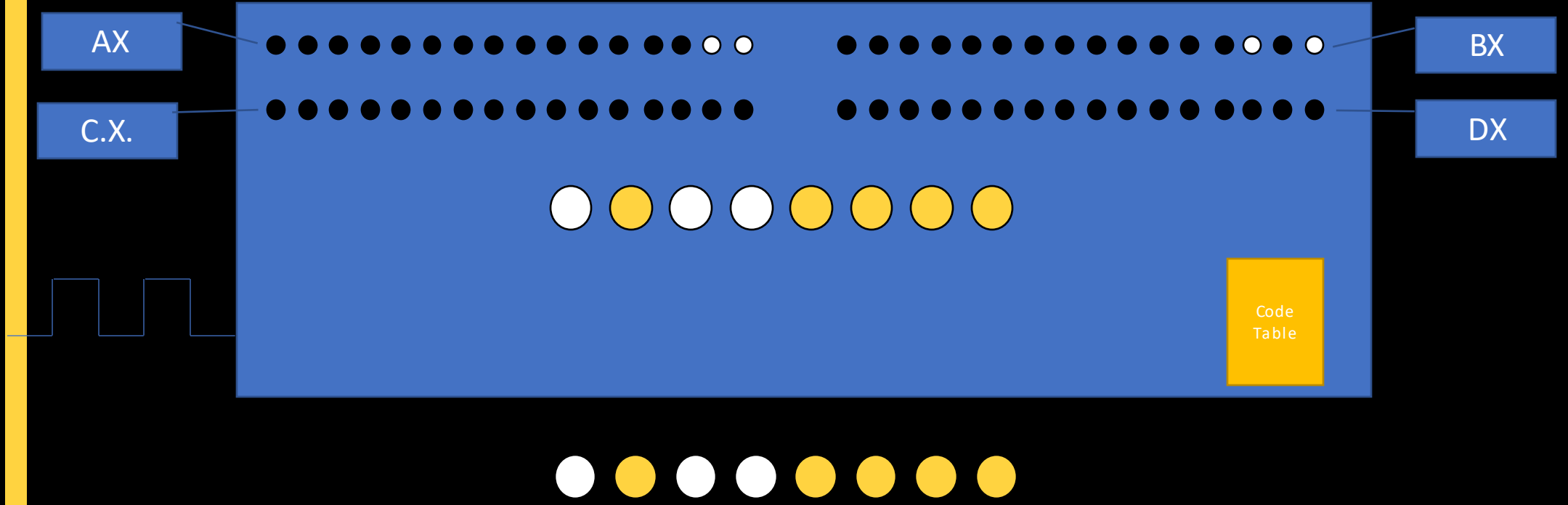


Central Processing Unit – Code Table

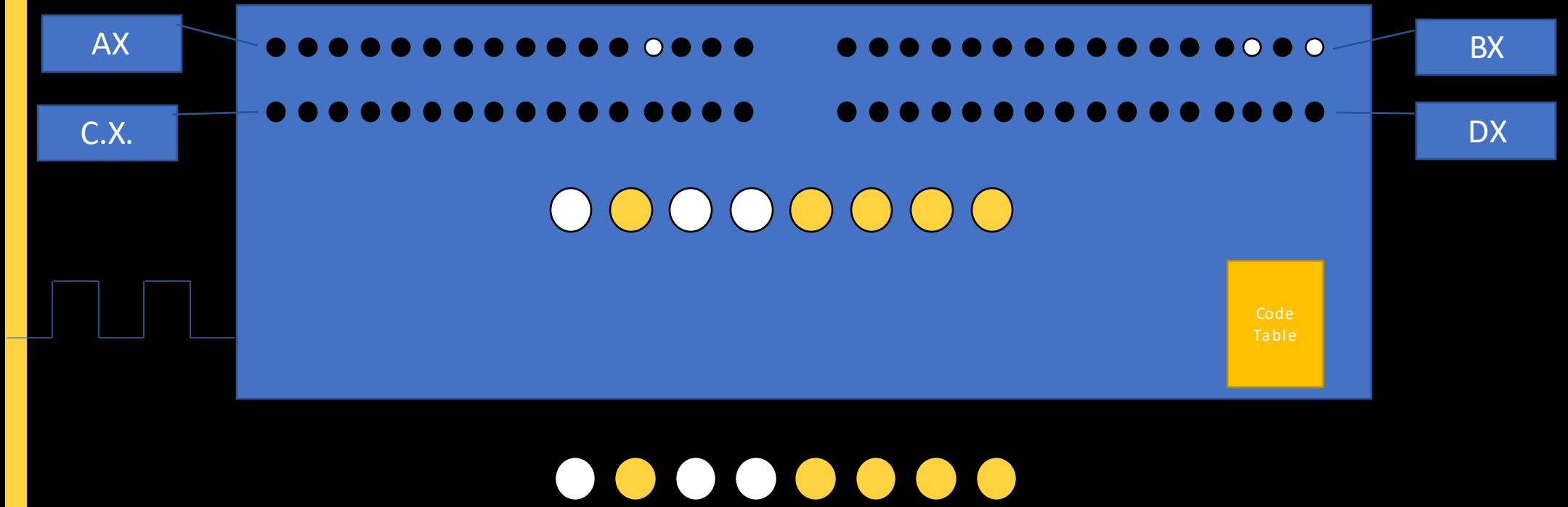


1000 0000	Next up is a number. Place it in the AX register.
1001 0000	Next up is a number. Place it in the BX register.
1011 0000	Add AX and BX, write the transaction to the AX register
1100 0000	Write the result to the external bus
0000 0000	0
0000 0001	1
0000 0010	2
0000 0011	3

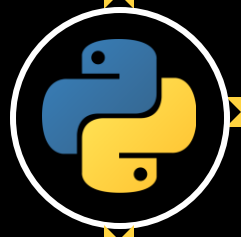
Central Processing Unit



Central Processing Unit

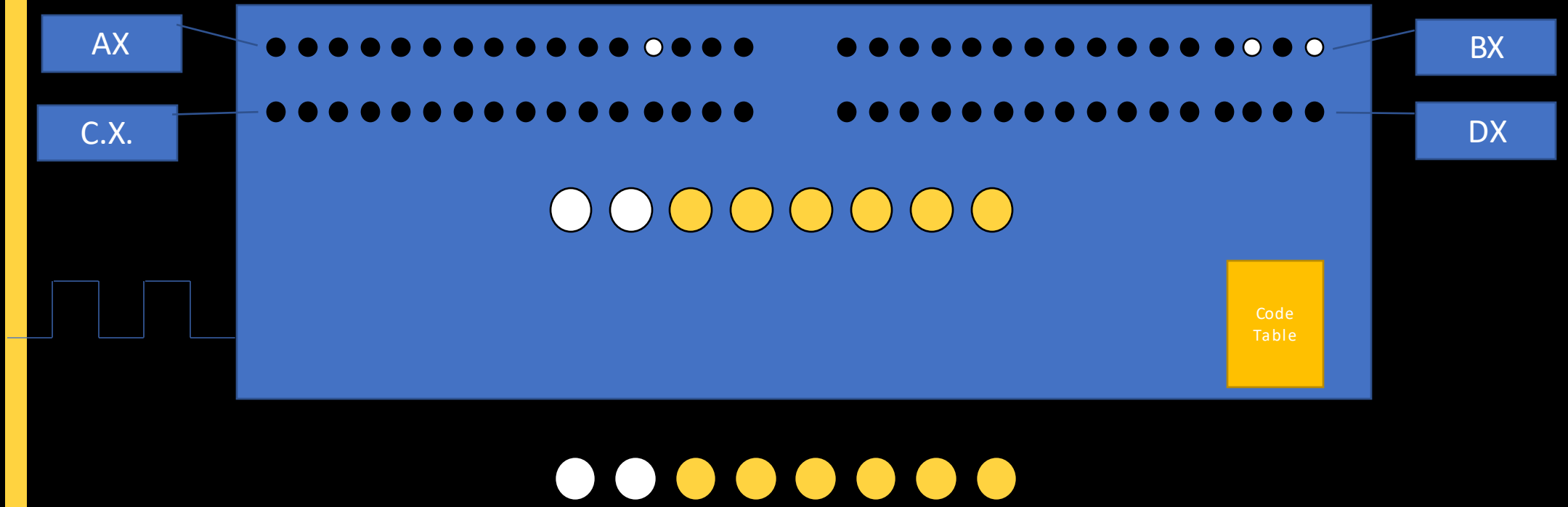


Central Processing Unit – Code Table

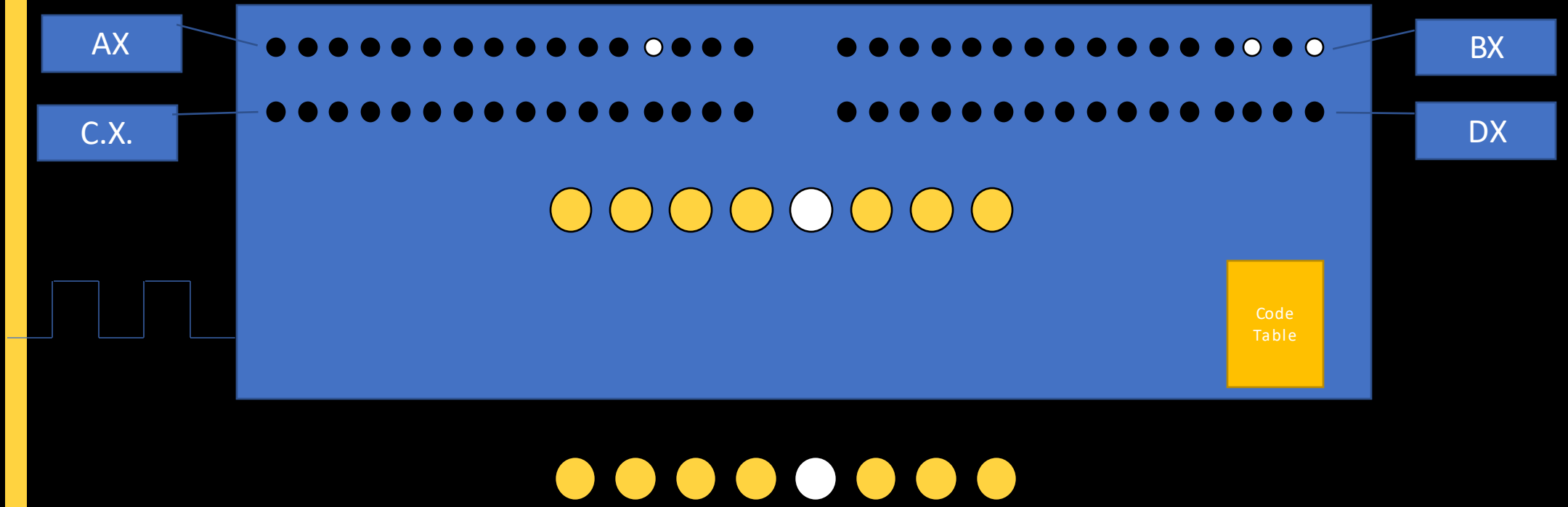


1000 0000	Next up is a number. Place it in the AX register.
1001 0000	Next up is a number. Place it in the BX register.
1011 0000	Add AX and BX, write the transaction to the AX register
1100 0000	Write the result to the external bus
0000 0000	0
0000 0001	1
0000 0010	2
0000 0011	3

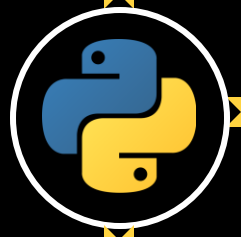
Central Processing Unit



Central Processing Unit

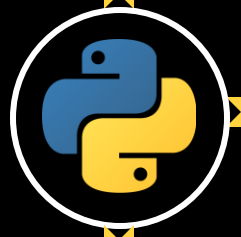


Compilation - Interpretation



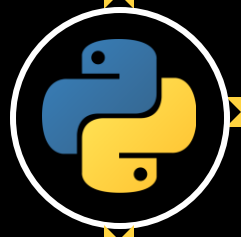
- COMPILE - the source program is translated once (however, this process must be repeated each time you change the source code) by taking a file containing the machine (for example, a .exe file if the code is intended to be run under MS Windows) Now you can distribute the file worldwide; The program that performs this translation is called a compiler or translator;

Compilation - Interpretation



- INTERPRETATION - you (or any user of the code) can interpret the source program each time it needs to be run; The program that performs this type of transformation is called an interpreter because it interprets the code each time it is intended to be run; This also means that you cannot distribute the source code as is, because the end user also needs the interpreter to execute it.

What Does the Interpreter Do?



- A computer program is essentially a piece of text, so its source code is usually placed in text files.
- The interpreter reads source code in a way common in Western culture: top-down and left-to-right.
- The interpreter checks if all subsequent lines are correct.
- When the interpreter finds an error, it immediately finishes its job. In this case the only result is an error message.
- If the line looks good, the interpreter tries to execute it.

Differences Between Compiler and Interpreter

	Compiler	Interpreter
Advantage	• Translated code is generally faster to execute; • Only the user needs to have the compiler - the end user can use the code without it; • The translated code is stored using machine language; Since it is so difficult to understand, your own inventions and programming tricks will most likely remain your secret.	• You can run the code as soon as it is completed; there are no additional stages of translation; • Code is stored using programming language, not machine language; This means that the code can be run on computers using different machine languages; You don't compile your code separately for each different architecture.
Disadvantage	• The compilation itself can be a very time-consuming process; you may not be able to run your code immediately after making changes; • You need to have as many compilers as there are hardware platforms you want your code to run on.	• Don't expect commenting to speed up your code; your code will be sharing the computer's power with the interpreter, so it can't be really fast; • Both you and the end user must have the interpreter to run your code.