```python
from PyQt5.QtWidgets import QTableWidgetItem, QMessageBox
from PyQt5.QtGui import QFont
from PyQt5.QtCore import Qt

def message_box_errors(errors: list):
    """
    Fenster, das alle Elemente aus errors anzeigt
    """
    info = QMessageBox()
    info.setIcon(QMessageBox.Critical)
    info.setWindowFlags(Qt.WindowTitleHint | Qt.CustomizeWindowHint | Qt.MSWindowsFixedSizeDialogHint)
    font = QFont()
    font.setPointSize(10)
    info.setFont(font)
    info.setText("\n".join(errors))
    info.setWindowTitle("Fehler")
    info.exec_()

class CU:  # Control Unit = Steuerwerk
    def __init__(self, ui, instruction_set, timer, cache, alu, input_register, output_register):
        self.ui = ui
        self.CACHE = cache
        self.ALU = alu
        self.input_register = input_register
        self.timer = timer
        self.output_register = output_register
        self.register = ["?", "0"]
        self.program_counter = 0
        self.instruction_set = instruction_set

    def fetch(self):
        if self.program_counter == -1:
            self.ui.label_which_cycle.setText("IDLE")
            return

        self.register[0] = self.CACHE.read(self.program_counter)
        self.register[1] = self.CACHE.read(self.program_counter + 1)

        self.ui.label_befehlszaehler.setText(str(self.program_counter))
        enc = encode_instruction(self.instruction_set, str(self.register[0]), str(self.register[1]))
        self.ui.label_befehlsreg_1.setText(enc[0])
        self.ui.label_befehlsreg_2.setText(enc[1])

    def decode(self):
        if self.program_counter == -1:
            self.ui.label_which_cycle.setText("IDLE")
            return
```

```
48      elif self.register[0] not in self.instruction_set:
49          self.program_counter = -1
50          self.ui.label_which_cycle.setText("IDLE")
51          return
52
53      self.ui.label_dekodierer_1.setText(str(self.register[0]))
54      self.ui.label_dekodierer_2.setText(str(self.register[1]))
55
56      if self.register[0] == self.instruction_set[0]:  # ld
57          if self.register[1].startswith("@"):
58              self.ALU.acc = self.CACHE.read(int(self.register[1].replace("@", "")))
59          else:
60              self.ALU.acc = self.register[1]
61
62      elif self.register[0] == self.instruction_set[1]:  # st
63          if self.register[1].startswith("@"):
64              self.CACHE.write(int(self.register[1].replace("@", "")), self.ALU.acc)
65              try:
66                  if int(self.register[1].replace("@", "")) % 2 == 0:
67                      self.ui.Speicher.setItem(int(int(self.register[1].replace("@", "")) / 2), 1, QTableWidgetItem(self.ALU.acc))
68                  else:
69                      self.ui.Speicher.setItem(int(int(self.register[1].replace("@", "")) / 2), 3, QTableWidgetItem(self.ALU.acc))
70              except Exception as e:
71                  print(e.__reduce_ex__(0))
72                  message_box_errors(["Ein Fehler ist aufgetreten!"])
73          else:
74              message_box_errors(["Ein Fehler ist aufgetreten!"])
75
76      elif self.register[0] == self.instruction_set[2]:  # in
77          if self.register[1].startswith("@"):
78              self.ui.input_reg.setEnabled(True)
79              self.ui.button_weiter.setEnabled(False)
80              self.ui.input_reg.setPlaceholderText(self.register[1])
81              self.ui.input_reg.setFocus()
82              self.timer.stop()  # input handled in Program.input_pressed
83          else:
84              message_box_errors(["Ein Fehler ist aufgetreten!"])
85
86      elif self.register[0] == self.instruction_set[3]:  # out
87          if self.register[1].startswith("@"):
88              self.output_register.text = self.CACHE.read(int(self.register[1].replace("@", "")))
89              self.ui.output_reg.setText(str(self.output_register.text))
90          else:
91              message_box_errors(["Ein Fehler ist aufgetreten!"])
92
93      elif self.register[0] == self.instruction_set[4]:  # add
94          if self.register[1].startswith("@"):
95              self.ALU.operand = self.CACHE.read(int(self.register[1].replace("@", "")))
```

```
96              else:
97                  self.ALU.operand = self.CACHE.read(int(self.program_counter + 1))
98              self.ALU.operation = "+"
99
100         elif self.register[0] == self.instruction_set[5]:  # sub
101             if self.register[1].startswith("@"):
102                 self.ALU.operand = self.CACHE.read(int(self.register[1].replace("@", "")))
103             else:
104                 self.ALU.operand = self.CACHE.read(self.program_counter + 1)
105             self.ALU.operation = "-"
106
107         elif self.register[0] == self.instruction_set[6]:  # mul
108             if self.register[1].startswith("@"):
109                 self.ALU.operand = self.CACHE.read(int(self.register[1].replace("@", "")))
110             else:
111                 self.ALU.operand = self.CACHE.read(self.program_counter + 1)
112             self.ALU.operation = "*"
113
114         elif self.register[0] == self.instruction_set[7]:  # div
115             if self.register[1].startswith("@"):
116                 self.ALU.operand = self.CACHE.read(int(self.register[1].replace("@", "")))
117             else:
118                 self.ALU.operand = self.CACHE.read(self.program_counter + 1)
119             self.ALU.operation = "/"
120
121         elif self.register[0] == self.instruction_set[8]:  # mod
122             if self.register[1].startswith("@"):
123                 self.ALU.operand = self.CACHE.read(int(self.register[1].replace("@", "")))
124             else:
125                 self.ALU.operand = self.CACHE.read(self.program_counter + 1)
126             self.ALU.operation = "%"
127
128         elif self.register[0] == self.instruction_set[9]:  # cmp
129             if self.register[1].startswith("@"):
130                 self.ALU.operand = self.CACHE.read(int(self.register[1].replace("@", "")))
131             else:
132                 self.ALU.operand = self.CACHE.read(self.program_counter + 1)
133             self.ALU.operation = "?"
134
135         if self.program_counter != -1:
136             self.program_counter += 2
137
138         if self.register[0] == self.instruction_set[10]:  # jmp
139             self.program_counter = int(self.CACHE.read(self.program_counter - 1)) * 2 - 2
140
141         elif self.register[0] == self.instruction_set[11]:  # jlt
142             if self.ALU.result == "-1":
143                 self.program_counter = int(self.CACHE.read(self.program_counter - 1)) * 2 - 2
```

```
144
145         elif self.register[0] == self.instruction_set[12]:  # jeq
146             if self.ALU.result == "0":
147                 self.program_counter = int(self.CACHE.read(self.program_counter - 1)) * 2 - 2
148
149         elif self.register[0] == self.instruction_set[13]:  # jgt
150             if self.ALU.result == "1":
151                 self.program_counter = int(self.CACHE.read(self.program_counter - 1)) * 2 - 2
152
153         if self.register[0] == self.instruction_set[14]:  # end
154             self.program_counter = -1
155             self.ui.label_which_cycle.setText("IDLE")
156
157     def execute(self):
158         if self.program_counter == -1:
159             self.ui.label_which_cycle.setText("IDLE")
160             return
161
162         if self.register[0] in self.instruction_set[4:10]:
163             self.ALU.compute()
164
165
166 def encode_instruction(instruction_set: list, instruction: str, val: str):
167     if len(instruction_set) != 15:
168         return "0", "0"
169
170     elif instruction == instruction_set[0]:  # ld
171         return "101" if str(val).startswith("@") else "100", str(val).replace("@", "")
172
173     elif instruction == instruction_set[1] and val.startswith("@"):  # st
174         return "111", str(val.replace("@", ""))
175
176     elif instruction == instruction_set[2] and val.startswith("@"):  # in
177          return "121", str(val.replace("@", ""))
178
179     elif instruction == instruction_set[3] and val.startswith("@"):  # out
180         return "131", str(val.replace("@", ""))
181
182     elif instruction == instruction_set[4]:  # add
183         return "201" if str(val).startswith("@") else "200", str(val).replace("@", "")
184
185     elif instruction == instruction_set[5]:  # sub
186         return "211" if str(val).startswith("@") else "210", str(val).replace("@", "")
187
188     elif instruction == instruction_set[6]:  # mul
189         return "221" if str(val).startswith("@") else "220", str(val).replace("@", "")
190
191     elif instruction == instruction_set[7]:  # div
```

```
192         return "231" if str(val).startswith("@") else "230", str(val).replace("@", "")
193
194     elif instruction == instruction_set[8]:  # mod
195         return "241" if str(val).startswith("@") else "240", str(val).replace("@", "")
196
197     elif instruction == instruction_set[9]:  # cmp
198         return "251" if str(val).startswith("@") else "250", str(val).replace("@", "")
199
200     elif instruction == instruction_set[10] and not val.startswith("@"):  # jmp
201         return "300", str(val)
202
203     elif instruction == instruction_set[11] and not val.startswith("@"):  # jlt
204         return "310", str(val)
205
206     elif instruction == instruction_set[12] and not val.startswith("@"):  # jeq
207         return "320", str(val)
208
209     elif instruction == instruction_set[13] and not val.startswith("@"):  # jgt
210         return "330", str(val)
211
212     elif instruction == instruction_set[14] and not val.startswith("@"):  # end
213         return "400", "0"
214
215     else:
216         return "0", "0"
```