```
FUNCTION_BLOCK INTERNAL FINAL cycle
VAR_INPUT
    EN : BOOL ;
END_VAR
VAR_OUTPUT
END_VAR
VAR
    // Single-scan tick
    Tick_0     : Tick ;            // Derives TICK one-shot from CLK
    RESET_ : BOOL ;
END_VAR

VAR
    RP_0 : Pulse_Relay ; // e.g., strobe Dispenser_0
    RP_1 : Pulse_Relay ; // e.g., strobe Turntable_0 (index)
    RP_2 : Pulse_Relay ; // e.g., strobe Press_Height_0 (start)
    RP_3 : Pulse_Relay ; // e.g., strobe Detector (capture)
    RP_4 : Pulse_Relay ; // e.g., strobe Decide_WP_0 (evaluate)
    RP_5 : Pulse_Relay ; // e.g., strobe Pickup (grab/reject chute gate)
END_VAR

VAR
    // Motion / actuators
    Dispenser_0      : Dispenser_module ;         // Places WP on station
    Turntable_0      : Turntable_module ;         // Indexes WP between positions
    Press_Height_0   : WP_size_identifier_module ;      // Sets/commands press height
    p                : Pick_and_drop_module ;            // Picks accepted WP

    // Sensing / logic
    Detector         : WP_color_identifier_module ;  // Reads color/size
    filter_wp_0      : WP_Filter ;         // Filters noisy detection
    Decide_WP_0      : WP_type_identifier ;         // Maps attributes → WP_TYPE decision

    TURNTABLE_DONE   : BOOL ;
    DISPENSER_DONE   : BOOL ;
    PRESSURE_DONE    : BOOL ;
    SENSOR_DONE      : BOOL ;
    DECIDE_DONE      : BOOL ;
    PICK_DROP_DONE   : BOOL ;
    RESET            : BOOL ;

    NOT_FOUND : BOOL ;
    SINK : BOOL ;
END_VAR

VAR CONSTANT
    FULL_CYCLE_PERIOD      : TIME := T#3.3S ;           // One complete station cycle
    DELAY_BEFORE_ROTATE    : TIME := T#500ms ;        // Time from dispense → rotate
    DELAY_BEFORE_DISPENSE  : TIME := T#2700MS ;       // Time from cycle start → dispense
    DELAY_BEFORE_PRESS     : TIME := T#2800MS ;       // Time from rotate → press
    DELAY_BEFORE_DETECT    : TIME := T#3000MS ;       // Time from press → detectd
    DELAY_BEFORE_DECIDE    : TIME := T#3100MS ;       // Time from detect → decide
    DELAY_BEFORE_PICKUP    : TIME := T#2800MS ;       // Time from decide → pickup

    COOLDOWN_TIME          : TIME := T#3s + T#1s ;    // Extra dwell at end of cycle
    COOLDOWN_RESET_TIME    : TIME := T#2s ;           // Time to hold RESET logic active
END_VAR
```