Comprehensive Technical Architecture for Overall Equipment Effectiveness (OEE) Calculation Systems in Microsoft Excel

Executive Summary

The modernization of manufacturing metrics is a critical step in the journey toward Industry 4.0, yet the complexity of implementing robust Manufacturing Execution Systems (MES) often leaves small to mid-sized enterprises (SMEs) relying on fragmented paper logs or ad-hoc spreadsheets. Overall Equipment Effectiveness (OEE) stands as the gold standard for measuring manufacturing productivity, identifying the percentage of manufacturing time that is truly productive. While the final output requested by stakeholders is often a "short and precise" Excel template, the engineering required to produce such a tool—one that is resilient to operator error, accurate in its handling of time losses, and scalable for longitudinal analysis—is anything but trivial.

This report provides an exhaustive analysis of the theoretical and practical construction of an OEE calculation engine within Microsoft Excel. It moves beyond simple cell arithmetic to discuss the database normalization required for tabular data entry, the conditional logic necessary to handle shift patterns and micro-stops, and the visualization techniques that convert raw integers into actionable operational intelligence. By dissecting the three pillars of OEE—Availability, Performance, and Quality—and mapping them to specific Excel functions and architectural decisions, this document serves as a definitive guide for process engineers and plant managers. The analysis synthesizes best practices from lean manufacturing theory with the technical capabilities of modern spreadsheet software, ensuring that the resulting tool satisfies the dual requirements of operational precision and user simplicity.

---

1. Introduction: The Intersection of Lean Metrics and Spreadsheet Logic

The quest for manufacturing efficiency is rooted in the elimination of waste, or *Muda*, a concept popularized by the Toyota Production System. OEE is the quantitative manifestation of this philosophy, distilling complex production dynamics into a single percentage that

represents the ratio of "Fully Productive Time" to "Planned Production Time." However, the translation of this theoretical metric into a functional digital tool requires a nuanced understanding of both manufacturing reality and software architecture.

1.1 The Operational Necessity of Digital OEE

In many production environments, OEE is calculated retrospectively, often days after the shift has ended. This latency destroys the metric's utility as a feedback loop for immediate continuous improvement (Kaizen). A manual calculation on a whiteboard or a calculator is prone to arithmetic errors and offers no historical traceability. Conversely, a full-scale MES implementation can cost hundreds of thousands of dollars and take months to deploy.

Microsoft Excel occupies a unique strategic niche in this landscape. It is ubiquitous, highly customizable, and capable of handling the data volume of a typical production line for extended periods. When architected correctly, an Excel-based OEE system provides a "short and precise" interface for the operator while performing complex data aggregation in the background. The goal is to reduce the friction of data entry to near zero, thereby ensuring high data integrity, while providing management with sophisticated, automated insights into the "Six Big Losses" of manufacturing.

1.2 The Scope of the "Short and Precise" Template

The user requirement for a "short and precise" template implies a design philosophy centered on **Minimal Viable Complexity**. The interface presented to the shop floor must be stripped of all extraneous elements. Operators should not be burdened with calculating "Run Time" or "Availability percentages"; they should strictly input what they observe—start times, stop times, and piece counts. The complexity must be completely encapsulated within the hidden calculation engine of the workbook.

This report will demonstrate that a "short" template is the result of a "long" design process. To make a template that requires only five inputs from a user, the architect must build a backend that handles unit conversions, error trapping, null value management, and conditional formatting. The precision of the output is directly proportional to the rigor of the input definitions and formulaic structures established during the design phase.

---

2. Theoretical Framework: The Anatomy of Loss and Excel Mapping

To build an accurate calculation engine, we must first rigorously define the variables. OEE is not merely a number; it is a taxonomy of time. Every minute of a 24-hour day must be accounted for and categorized correctly in the Excel logic to prevent "leakage"—where time

disappears from the record, artificially inflating performance scores.

2.1 The Hierarchy of Time and the Six Big Losses

The foundation of OEE lies in the subtraction of losses from the total available time. This hierarchy must be reflected in the column structure of the Excel database.

Total Operations Time (24/7)
This is the starting point. For a single shift, this is the Shift Length (e.g., 480 minutes for an 8-hour shift).
Level 1: Planned Production Time
We subtract Schedule Loss (no production scheduled, holidays, lack of orders).

- *Excel Implication:* The template must allow for the definition of "Non-Working Time." If a machine is not staffed, it should not penalize the OEE score. This is a critical distinction between OEE and TEEP (Total Effective Equipment Performance). OEE measures efficiency *when scheduled to run*; TEEP measures efficiency against *calendar time* (24/7/365).

Level 2: Run Time (Operating Time)
From Planned Production Time, we subtract Availability Losses. These are events that stop production for a measurable period (typically > 5 minutes).

- *Loss 1: Equipment Failure:* Unplanned downtime (breakdowns, tool breakage).
- *Loss 2: Setup and Adjustments:* Planned downtime (changeovers, material changes, warm-up).
- *Excel Implication:* The "Downtime" input column in Excel must capture both the *duration* and the *reason code*. While the basic calculation only needs the duration to determine Availability, the *insight* requires the reason.

Level 3: Net Run Time
From Run Time, we subtract Performance Losses. These are subtle losses where the machine is running, but not at full capacity.

- *Loss 3: Idling and Minor Stops:* Small stops (< 5 minutes) that are often unrecorded by operators (jams, sensor blocks).
- *Loss 4: Reduced Speed:* The machine is running at 80% of its nameplate capacity.
- *Excel Implication:* These losses are rarely entered manually. They are calculated variables. If the machine ran for 400 minutes and produced 400 parts (Ideal Cycle = 0.5 min), the theoretical time to make those parts was 200 minutes. The other 200 minutes were lost to Performance. The Excel formula must derive this gap automatically.[1]

Level 4: Fully Productive Time
From Net Run Time, we subtract Quality Losses.

- *Loss 5: Process Defects:* Parts scrapped during steady-state production.
- *Loss 6: Reduced Yield:* Parts scrapped during startup or changeover.
- *Excel Implication:* The "Count" columns must separate "Total Processed" from "Good

Output." A common error in Excel templates is asking only for "Good Count." This makes it impossible to calculate the Quality percentage accurately.[1]

3. Excel Architecture for Manufacturing Data

To deliver a "short and precise" template, we must decouple the user interface from the data storage and the calculation logic. A monolithic spreadsheet where operators type directly into formula cells is a recipe for disaster. We will employ a **Model-View-Controller (MVC)** adaptation suitable for Excel.

3.1 The Three-Sheet Logic Structure

The workbook should be composed of at least three distinct worksheets, each with a specific purpose and protection level.

Sheet 1: The "Settings" (The Constants)

This sheet functions as the system configuration file. It holds the variables that are constant across shifts but may change over months or years. By centralizing these, we avoid hard-coding numbers like "480 minutes" into thousands of row formulas.

Key Data Points for Settings:

- **Standard Shift Length (min):** e.g., 480.
- **Planned Break Time (min):** e.g., 30 (Lunch) + 15 (Break) = 45.
- **OEE Targets:** e.g., Availability 90%, Performance 95%, Quality 99%, OEE 85%.[8]
- **Machine Nameplate Speeds:** If multiple machines are being tracked, a lookup table here defines the Ideal Cycle Time for each machine or part number.

Excel Best Practice: Name these cells using Excel's "Name Manager" (e.g., ShiftLength, BreakTime). This makes formulas readable. Instead of =B2-C5, the formula becomes =ShiftLength - BreakTime.

Sheet 2: The "Data Log" (The Database)

This is the heart of the system. It is a structured table where each row represents a distinct production event or shift.

- **Structure:** Flat file database format.
- **Headers:** Date, Shift, Machine, Part Number, Inputs, Calculations.
- **Format:** Excel Table (Ctrl+T). This is non-negotiable. Excel Tables automatically expand formatting and formulas to new rows, ensuring the template doesn't "break" when the operator reaches row 100.

Sheet 3: The "Dashboard" (The Output)

This sheet contains no data entry. It is a canvas of Pivot Tables, Slicers, and Charts that read

from the Data Log. It provides the "Short and Precise" visual summary management requires.

3.2 Database Normalization and Granularity

A critical design decision is the granularity of time. Should one row represent an entire 8-hour shift, or should one row represent one hour, or one specific job?

Scenario A: The "Shift-Level" Template

- **Pros:** Shortest, simplest. The operator fills in one row at the end of the day.
- **Cons:** Low resolution. If OEE was low, you don't know *when* in the shift it happened. You only know the total downtime sum.
- **Verdict:** Best for the "Short and Precise" request for manual entry environments.

Scenario B: The "Job-Level" Template

- **Pros:** High accuracy for high-mix environments. New row for every product changeover.
- **Cons:** Higher data entry burden.
- **Verdict:** Necessary if "Ideal Cycle Time" varies significantly between products. If Product A takes 10 seconds and Product B takes 60 seconds, averaging them in a single shift row destroys the Performance accuracy.

For the purpose of this report, we will focus on a hybrid **Shift-Level** design that includes columns for product mix, or assumes a dominant product, as this aligns best with the user's request for simplicity while maintaining mathematical rigor.

---

4. Designing the User Interface (The Input Layer)

The "Input Section" is the only part of the spreadsheet the operator interacts with. Its design determines the data quality. If it is confusing, operators will enter bad data ("GIGO" - Garbage In, Garbage Out). The interface must be constrained and validated.

4.1 Column Definitions and Data Validation

We will define the specific columns for the **Data Log** sheet.

Column	Header Name	Data Type	Validation Source	Description
A	**Date**	Date	Date (not future)	The production date.

B	**Shift**	Text	List: 1, 2, 3	The working shift.
C	**Machine ID**	Text	List: Settings!MachineList	Dropdown menu of assets.
D	**Part Number**	Text	List: Settings!PartList	Dropdown of active products.
E	**Ideal Cycle (s)**	Number	Positive Decimal	**Critical:** Theoretical sec/part.[6]
F	**Total Pieces**	Number	Whole Number	Total output (Good + Bad).
G	**Reject Pieces**	Number	Whole Number <= Col F	Bad parts. Cannot exceed Total.
H	**Downtime (min)**	Number	Decimal <= ShiftLength	Total unplanned stop time.
I	**Comments**	Text	Free Text	Context for losses.

The Importance of Dropdown Lists (Data Validation):
Using Data > Data Validation > List is essential for columns like "Machine ID" and "Part Number."

- *Reasoning:* If Operator A types "Machine 1" and Operator B types "Mach 1", the Pivot Table will see them as two different machines, breaking the analysis. Consistency is key for the "precise" aspect of the user request.

4.2 Handling "Ideal Cycle Time" - The Achilles Heel

The most common point of failure in OEE templates is the **Ideal Cycle Time**.

- *The Problem:* Operators often don't know the exact theoretical cycle time (e.g., 4.5 seconds). They might guess, or type in the "Goal" (which includes allowances). This

corrupts the Performance calculation.
- *The Precise Solution:* Do not let operators type this. Use a VLOOKUP or XLOOKUP formula in Column E that references the **Part Number** in Column D.
    - Formula in E2: =XLOOKUP([@[Part Number]], Settings!$A:$A, Settings!$B:$B, "Check Part #")
    - This ensures that the standard is always applied consistently. The operator only selects "Part Number"; the spreadsheet supplies the physics.

4.3 Input Sanitization and Error Trapping

Humans make typos. An operator might accidentally type "4800" instead of "480" for downtime.

- **Constraint:** Apply Data Validation to the "Downtime" column.
    - *Rule:* Whole Number between 0 and ShiftLength (e.g., 480).
    - *Error Message:* "Stop! Downtime cannot exceed the length of the shift."
- **Constraint:** Apply Data Validation to "Rejects."
    - *Rule:* Less than or equal to "Total Pieces."
    - *Error Message:* "Rejects cannot differ from Total Production."

These constraints enforce the "Precision" requested by the user, preventing mathematically impossible scenarios from entering the system [5]

---

5. The Calculation Engine: Formulaic Derivation

Once the data is safely entered into Columns A through H, the "Calculation Engine" (Columns I through P) takes over. These columns should ideally be locked or hidden to prevent tampering. We will derive these formulas step-by-step, addressing the nuances of Excel syntax and time units.

5.1 Calculating Planned Production Time (The Denominator)

Before we calculate Availability, we must establish the baseline.

- **Concept:** Shift Length minus Statutory Breaks.
- **Excel Formula (Col J):**
  Excel
  `= -`

  Note: We assume ShiftLength and PlannedBreaks are pulled from the Settings sheet, or are columns if they vary. If using the Named Range approach from Section 3.1:
  Excel
  `=Settings!StandardShift - Settings!StandardBreaks`

- **Context:** This represents the maximum time the machine *could* run if nothing went wrong. If a machine is scheduled for 8 hours but has a 30-minute unpaid lunch and two 10-minute breaks, the Planned Production Time is $480 - 50 = 430$ minutes.[4]

5.2 Calculating Availability (The Uptime Metric)

- **Concept:** (Planned Production Time - Unplanned Downtime) / Planned Production Time.
- **Excel Formula (Col K - Run Time):**
  Excel
  = -]

- **Excel Formula (Col L - Availability %):**
  Excel
  =IF(=0, 0,] /)

- **Analysis:** We use the IF statement to handle the "Division by Zero" error. If Planned Time is zero (a non-scheduled day), the Availability is 0 (or blank), not #DIV/0!. This ensures the report looks clean ("Short and Precise").
- **Source Validation:** This aligns with the standard availability definition found in snippet [2] and.[4]

5.3 Calculating Performance (The Speed Metric)

This is the most mathematically sensitive section due to unit mixing (seconds vs minutes).

- **Concept:** (Total Parts * Ideal Cycle Time) / Run Time.
- **Unit Check:**
  - Run Time is in **Minutes**.
  - Ideal Cycle Time is usually in **Seconds**.
  - We must convert one to match the other. Standard practice is to convert everything to Minutes.
- Intermediate Calculation (Col M - Theoretical Time):
  This is the time it should have taken to make the parts.
  Excel
  =() * [@[Ideal Cycle (s)]]) / 60

- **Excel Formula (Col N - Performance %):**
  Excel
  =IF(]=0, 0,] /])

- **Performance Cap:** Sometimes, Performance > 100% (e.g., 105%). This implies the cycle time standard is wrong (too conservative). Some templates cap this at 100% using =MIN(1, calculation), but it is better to leave it uncapped to reveal the bad standard. A value of 120% highlights a need to update the Ideal Cycle Time in the Settings tab.. [5]

5.4 Calculating Quality (The Yield Metric)

- **Concept:** Good Parts / Total Parts.
- **Intermediate Calculation (Col O - Good Pieces):**
  Excel
  `=] -]`

- **Excel Formula (Col P - Quality %):**
  Excel
  `=IF(]=0, 0, [@[Good Pieces]] /])`

- **Insight:** This logic penalizes rework. If a part is made, rejected, fixed, and shipped, strict OEE counts it as a loss on the first pass. This formula reflects First Pass Yield (FPY).[1]

5.5 The Master OEE Formula

- **Concept:** Availability * Performance * Quality.
- **Excel Formula (Col Q - OEE% %):**
  Excel
  `=[@[Availability %]] * [@[Performance %]] * [@[Quality %]]`

- **Alternative Audit Formula:**
  Excel
  `=IF(=0, 0,] * [@[Quality %]] /)`

  (Note: This simplifies to Good Time / Planned Time).

By placing these formulas in an Excel Table, they automatically propagate. The user inputs data in A-H, and columns I-Q populate instantly. This fulfills the requirement for a "precise" automated tool.

---

6. Advanced Data Handling and Sanitization

To make the template "Short and Precise," we must hide the complexity of real-world messiness. Data is rarely clean. Operators forget to log entries; shifts run over midnight; machines run without scheduled operators. A robust Excel architecture must handle these edge cases without crashing.

6.1 Handling Midnight Crossover

A common headache in manufacturing tracking is the "Third Shift" problem. A shift starts at 10:00 PM and ends at 6:00 AM the next day.

- *The Issue:* If you calculate duration using =EndTime - StartTime, Excel sees 06:00 - 22:00 as a negative number.
- *The Precise Solution:* Use the modulo function or standard date math.
    - Formula: =MOD(EndTime - StartTime, 1)
    - This assumes the end time is the next day if it's smaller than the start time.
    - *Recommendation:* In the "Short" template, avoid "Start/End Time" inputs entirely. Ask for "Duration in Minutes" or "Shift Name." This bypasses the date math complexity completely, adhering to the principle of simplicity.

6.2 Weighted Averages for Aggregate Reporting

When the user asks for a template, they likely want to see a weekly or monthly summary. **Do not average the OEE column.**

- *The Math:* Average(80%, 90%) = 85%. But if the 80% shift was 12 hours long and the 90% shift was 1 hour long, the true average is much closer to 80%.
- *The Excel Solution:* We must calculate **Weighted OEE**.
    - This creates a requirement for "Helper Columns" in our table.
    - We need sumable numerators and denominators.
    - **Sum_Available_Time** (Run Time)
    - **Sum_Planned_Time**
    - **Sum_Theoretical_Time**
    - **Sum_Good_Parts**
- *Pivot Table Calculation:* Instead of dragging "OEE" to the values area and selecting "Average", we create a **Calculated Field** in the Pivot Table:
    - Weighted OEE = 'Theoretical Time' / 'Planned Production Time' * 'Quality Rate' (simplified).
    - More accurately: We sum the *time losses* and subtract from the total.
    - Ideally: $\frac{\sum(Good Count \times Ideal Cycle)}{\sum(Planned Production Time)}$
    - This single formula in a Pivot Table provides the mathematically correct OEE for any time range selected (Day, Week, Month, Year).[3]

6.3 Handling "Null" vs "Zero."

In Excel, a blank cell is treated as a zero in some math operations but ignored in others (like AVERAGE).

- *Scenario:* A machine is down for maintenance all day (No scheduled production).
- *Input:* Shift Length = 0.
- *Result:* OEE formulas return errors or zeros.
- *Fix:* The IFERROR wrapper used in Section 5 handles the error. However, for charting, we don't want a "0%" point on the line chart (which looks like a failure); we want a *gap* (missing data).
- *Excel Tip:* In the formula, use NA() instead of 0.

- =IF(=0, NA(),...)
- Excel Charts ignore #N/A errors and interpolate the line, keeping the visual trend clean ("Short and Precise" visualization).

---

7. Visualization and Dashboarding

The "Short and Precise" requirement is most critical in the output. A manager wants to glance at the screen and know the status in 5 seconds. This requires a dashboard that utilizes "Management by Exception."

7.1 Conditional Formatting as a Signaling System

We will apply a "Traffic Light" system to the OEE column. This is not just aesthetic; it guides the eye to actionable data.

The Rules (Standard Benchmarks):

1. **Red (Critical):** OEE < 60%. (Typical for unoptimized plants [8]).
2. **Yellow (Warning):** 60% <= OEE < 85%.
3. **Green (World Class):** OEE >= 85%.

Excel Implementation:

- Select the OEE column (Column Q).
- Home > Conditional Formatting > Icon Sets > 3 Signs.
- *Crucial Step:* Go to Manage Rules > Edit Rule. Change the "Type" from Percent to **Number**. Set Green >= 0.85 and Yellow >= 0.60. (Excel defaults to "Percent," which means "Top 33% of data", not "Mathematical 85%").

7.2 The Dynamic "Waterfall" Chart for Losses

A standard OEE number tells you *how* you did, but not *why*. A Waterfall chart (or Stacked Bar) is the precise tool to show where the time went.

Chart Structure:

- **Total Bar:** 100% (Planned Time).
- **Series 1 (Bottom):** Fully Productive Time (Green).
- **Series 2:** Quality Loss Time (Red).
- **Series 3:** Performance Loss Time (Orange).
- **Series 4:** Availability Loss Time (Blue).
- **Series 5 (Invisible):** Schedule Loss (if calculating TEEP).

Deriving the Data:
We need to calculate these time buckets in minutes in our Calculation Engine:

- Availability Loss = Planned Time - Run Time
- Performance Loss = Run Time - Theoretical Time
- Quality Loss = Theoretical Time - Fully Productive Time

By stacking these in a 100% Stacked Bar Chart, the manager sees a visual representation of the "Six Big Losses" for each day. If the Blue bar is huge, focus on maintenance. If the Orange bar is huge, focus on cycle times and micro-stops. This delivers high-density insight in a concise format.

7.3 The Pivot Dashboard

The dashboard sheet should contain:

1. **Slicer (Time):** Buttons for "This Month", "Last Month".
2. **Slicer (Machine):** Buttons for "Machine A", "Machine B".
3. **KPI Cards:** Large text boxes linked to cells showing the Weighted OEE for the selection.
4. **Trend Chart:** Line chart of OEE over time.
5. **Pareto Chart:** Bar chart of "Downtime Reasons" (if reason codes are captured).

This layout satisfies the "Short and Precise" brief: interactive, clean, and focused on key metrics.

---

8. Automating with VBA and Macros

While formulas handle the logic, Visual Basic for Applications (VBA) handles the *workflow*. To make the template robust enough for daily use, we can use simple scripts to automate repetitive tasks. This transforms a static spreadsheet into a mini-application.[3]

8.1 The "Reset Form" Macro

If operators are using a "Form View" (a specific input sheet) rather than typing into the database directly, we need a button to clear the inputs for the next shift.

VBA

```
Sub ClearForm()
    ' Purpose: Clears the data entry cells for the next user
    Dim ws As Worksheet
    Set ws = ThisWorkbook.Sheets("Input")
```

```
    ' Only clear the input cells, leave formulas alone
    ws.Range("B4:B10").ClearContents

    MsgBox "Form Cleared. Ready for next shift.", vbInformation
End Sub
```

8.2 The "Archive Data" Macro

As the Data Log grows, Excel slows down. A best practice for long-term templates is to have an "Archive" function that moves old data to a separate file or static sheet.

VBA

```
Sub ArchiveData()
    ' Purpose: Moves data older than 30 days to the Archive sheet
    Dim wsData As Worksheet, wsArch As Worksheet
    Dim lastRow As Long, i As Long.
    Dim cutDate As Date

    Set wsData = ThisWorkbook.Sheets("Data Log")
    Set wsArch = ThisWorkbook.Sheets("Archive")
    cutDate = Date - 30 ' 30 days ago

    Application.ScreenUpdating = False

    ' Loop backwards to avoid index errors when deleting row.s
    lastRow = wsData.Cells(wsData.Rows.Count, "A").End(xlUp).Row
    For i = lastRow To 2 Step -1
        If wsData.Cells(i, 1).Value < cutDate Then
            wsData.Rows(i).Copy Destination:=wsArch.Cells(wsArch.Rows.Count,
"A").End(xlUp).Offset(1)
            wsData.Rows(i).Delete
        End If
    Next i

    Application.ScreenUpdating = True
    MsgBox "Archiving Complete.", vbInformation
End Sub
```

- *Relevance:* This script ensures the "Short and Precise" template stays fast and

responsive, preventing the "bloat" that plagues complex Excel sheets over time.

8.3 Connecting to External Data (The Zero-Entry Goal)

The ultimate "Short" template is one that the operator doesn't touch at all. If the machine has a PLC (Programmable Logic Controller) that outputs a CSV file, we can use **Power Query** (Get & Transform) to suck that data in.

- **Method:** Data > Get Data > From Text/CSV.
- **Transformation:** Use the Power Query Editor to strip headers, filter nulls, and format columns.
- **Automation:** Set the query to "Refresh every 60 minutes."
- **Result:** The OEE Dashboard updates itself. The operator's job shifts from "Data Entry" to "Data Verification."

---

9. Strategic Implementation and Change Management

The best Excel template will fail if the human element is ignored. OEE is often weaponized by management as a tool to punish operators for downtime. This leads to data falsification (e.g., hiding downtime as "Part Changeover" to protect Availability scores).

9.1 The Psychology of Data Entry

To ensure the template works:

1. **Involve Operators in Design:** Ask them, "What is the hardest part of filling out the current log?" If they say "calculating the minutes," ensure the Excel template does that for them.
2. **Focus on "Loss" not "Blame":** Frame OEE as a tool to identify where the *machine* is failing the operator, not vice versa.
3. **The "Two-Minute Rule":** Data entry should take no more than two minutes per shift. The "Short and Precise" template design (Dropdowns, minimal fields) supports this goal.

9.2 From Excel to the Future

When does the organization outgrow this template?

- **Trigger 1:** Data volume exceeds 100,000 rows.
- **Trigger 2:** Multi-user collisions (two people trying to edit the file at once).
- **Trigger 3:** Need for real-time (sub-second) visibility.

At this point, the Excel logical architecture (the definitions of Availability, Performance, Quality) serves as the "Spec Sheet" for implementing a SQL database or a commercial MES. The Excel phase is a crucial prototyping ground where the organization learns what metrics

actually matter before spending capital on software.

---

10. Conclusion

To fulfill the user's request for an OEE Excel template that is "Short and Precise," we have paradoxically traveled a path of rigorous detail. We have established that simplicity in the user interface is achieved only through complexity in the architectural backend.

The Final Deliverable Specification:
A robust, professional OEE template consists of:

1. **Three Sheets:** Settings (Constants), Data Log (Table), Dashboard (Pivot).
2. **Seven User Inputs:** Date, Shift, Machine, Part #, Total Count, Rejects, Downtime Minutes.
3. **Three Lookup Values:** Ideal Cycle Time, Shift Length, Planned Breaks (Automated).
4. **Four Calculated Metrics:** Availability, Performance, Quality, OEE.
5. **One Visual Output:** A traffic-light-coded dashboard showing the trend.

By adhering to the formulas derived in Section 5—specifically handling unit conversions for Performance and error-trapping for Availability—the resulting tool will provide "Exhaustive" insight through a "Short" interface. This report empowers the user to build not just a spreadsheet, but a sustainable system for operational excellence, bridging the gap between lean theory and shop floor reality.