TRAINING MODULE

Yamazumi Chart

Imbalance a supervisor can read in three seconds

A working module for engineers with 1–2 years on the floor. By the end you will be able to read a stacked Yamazumi bar element by element, and calculate the proposed-state chart by hand using the same ECRS math the tool uses.

Grounded in Toyota's visual-management tradition and the classical ECRS (Eliminate, Combine, Rearrange, Simplify) method-study questions

BEFORE WE START

What You Will Be Able to Do After This Module

1. Explain what “Yamazumi” means and why a stacked bar communicates imbalance faster than a table of numbers.
2. Read every segment of a station's bar — element, time, and ECRS color — and total it against Takt.
3. Apply the exact ECRS saving formula the tool uses to hand-calculate a proposed-state bar.
4. Explain how an element with no manual ECRS tag can still get one — the MTM-inferred fallback.
5. Recognize the two most common mistakes junior IEs make when they first read a Yamazumi chart.

“Yamazumi” and the ECRS Question Set

Two ideas stack together in this one chart — know both before you read your first bar.

積む (YAMAZUMI)

Literally “to stack up.” On the shop floor, magnetic cards stacked on a board — one per work element — so a supervisor sees imbalance without reading a single number.

This tool is that physical board, generated from your real Production Study data.

ECRS

The four classical method-study questions, asked of every element: can it be Eliminated, Combined, Rearranged, or Simplified?

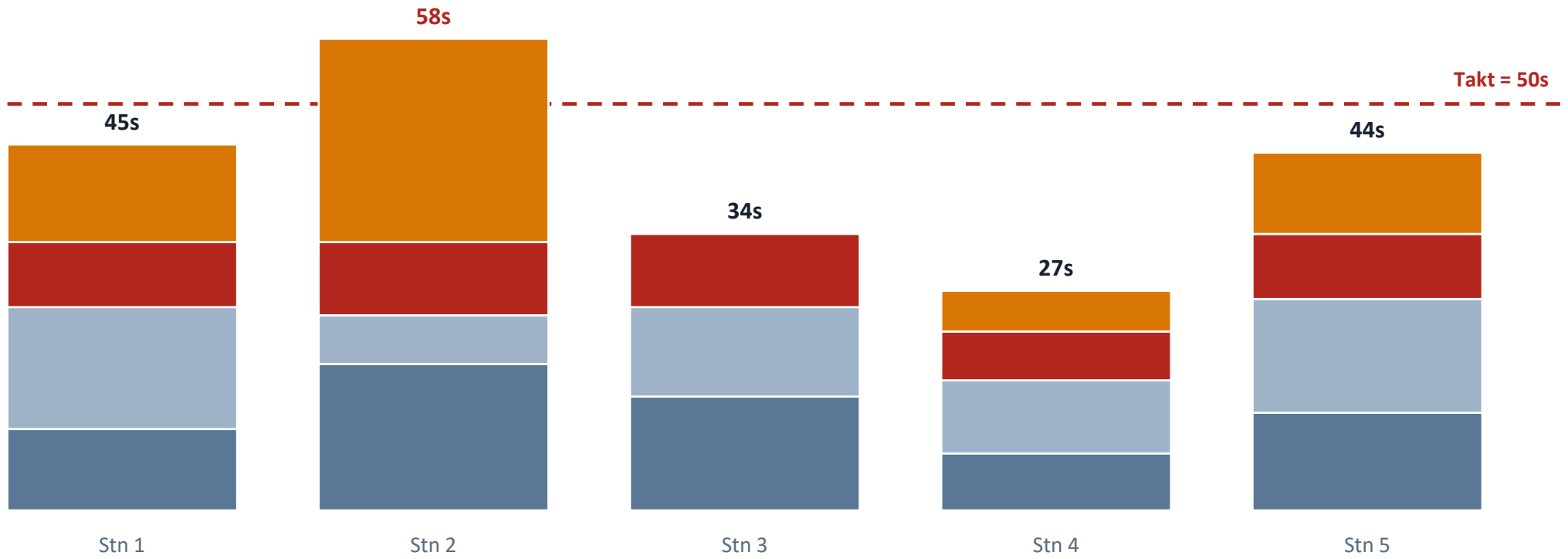
Every segment on the chart is color-coded by which of these four — or none — applies to it.

A Yamazumi chart is ECRS made visible, station by station, against a single Takt line drawn across every bar.

THE CORE IDEA

One Stacked Bar Per Station — Nothing Hidden in an Average

Five stations. Every bar shows its actual elements, stacked, colored, against one Takt line. No station can hide behind a tidy total anymore.



Station 2's 58s bottleneck isn't one number — it's four visible elements. You can already guess which segment to question first, before opening a single report.

WHERE THIS METHOD COMES FROM

Make the Problem Visible, and It's Already Half-Solved

Toyota's shop floors run on visual management: andon lights, kanban cards, and physical Yamazumi boards — all built on the same principle. A table of numbers needs interpretation. A stacked bar against a line needs none. The person walking the line doesn't have to be an IE to see which station is overloaded and which segment is the biggest piece of it.

This tool digitizes that board, without losing what made the physical version work.

VISUAL MANAGEMENT

Toyota Production System

READ THE LEGEND FIRST

Every Segment's Color Is a Claim About That Element

ELIMINATE	No value-add — candidate for full removal. The strongest claim on the chart.
COMBINE	Could merge with another element or station — partial time saving.
REARRANGE	Right work, wrong sequence position — partial time saving from resequencing.
SIMPLIFY	Necessary, but the method itself can be leaner — partial time saving.
PLAIN VA (no tag)	Core value-add work, on standard — stays exactly as is.

THE METHOD, IN ONE PICTURE

Three Layers Behind Every Chart

1

PRESENT STATE

Real element times, real ECRS tags, stacked exactly as recorded.

2

MTM-INFERRED FALLBACK

Untagged elements the MTM engine already flagged EXCEED get an inferred category — traceably marked.

3

PROPOSED STATE

The same saving formula fill_index.py already uses — this chart never disagrees with the Index sheet.

The rest of this module walks all three layers with ONE station, worked by hand.

1

LAYER 1

Reading a Present-State Bar

Five real elements, five real tags. Read every segment before you touch any math.

Notice E5 has no manual tag at all — that's the case Layer 2 exists for.

STATION 3 — FIVE ELEMENTS, TAKT = 50s

Elem	Time	ECRS Tag	Color
E1	12s	Eliminate	■ red
E2	18s	— (plain VA)	■ steel
E3	9s	Combine	■ amber
E4	14s	Rearrange	■ purple
E5	7s	no tag — MTM flagged EXCEED	■ blue (inferred)

Present-State Total = 12+18+9+14+7 = 60s — 10s over the 50s Takt. This station is a bottleneck, and now you can see exactly which five pieces make it one.

2

LAYER 2

MTM-Inferred Fallback

An untagged element isn't automatically “fine.” If MTM's own Gap/Flag logic already called it EXCEED, this chart doesn't stay silent about it.

An on-standard or core Use(U) element is NEVER auto-tagged — only a genuine EXCEED gets this treatment.

Layer 2 — MTM-Inferred Fallback

HOW E5 GOT ITS COLOR WITHOUT A MANUAL TAG

1. E5 carries no manual ECRS tick — nobody checked a box for it.
2. MTM RCA's own Gap/Flag logic already marked E5 as EXCEED (see the MTM module).
3. THERBLIG_META maps E5's Therblig category to an ECRS category — here, Simplify.
4. On a combo tag like “E/R”, priority order E > C > R > S picks the strongest claim.
5. The segment renders in its inferred color, marked “MTM-inferred” in the tooltip — traceable, never silent.

Why this matters: without this fallback, an untagged-but-genuinely-slow element would sit on the chart looking like “plain VA, nothing to see here” — exactly wrong.

3

LAYER 3

Proposed-State Math

The exact same formula fill_index.py already uses — memorize these three rules and you can hand-check any proposed bar.

Un-tagged, on-standard plain VA elements are never touched — they're already doing what they should.

THE THREE RULES

Eliminate → element time fully removed (−100%)

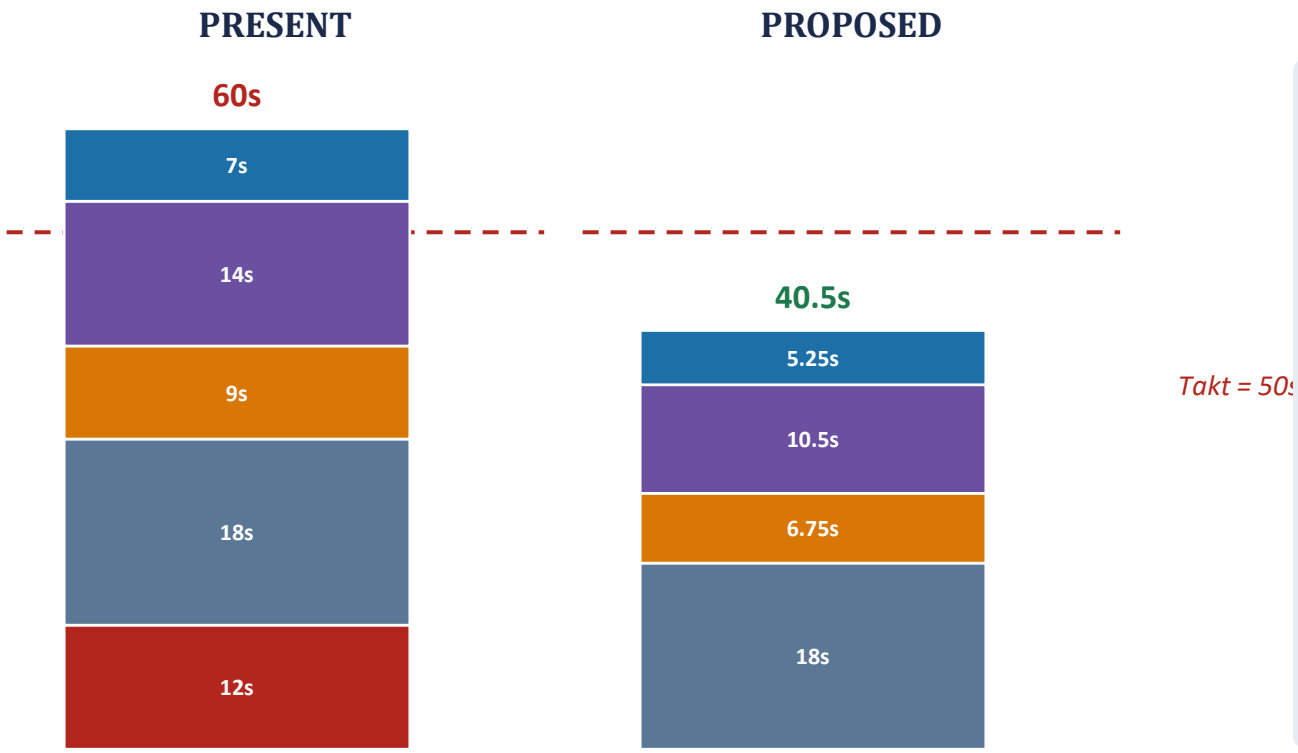
Combine / Rearrange / Simplify → element time reduced by 25%

Un-tagged, plain VA → element time unchanged (0%)

APPLIED TO STATION 3'S FIVE ELEMENTS

Elem	Present	Rule	Proposed
E1	12s	Eliminate → −100%	0s
E2	18s	no tag → unchanged	18s
E3	9s	Combine → −25%	6.75s
E4	14s	Rearrange → −25%	10.5s
E5	7s	Simplify (inferred) → −25%	5.25s

Station 3 — Present vs. Proposed, Side by Side



WHAT MOVED

- E1 Eliminate: 12s → 0s
- E2 unchanged: 18s
- E3 Combine: 9 → 6.75s
- E4 Rearrange: 14 → 10.5s
- E5 Simplify: 7 → 5.25s

60s → 40.5s
Under Takt by 9.5s

BEFORE YOU GO TO THE FLOOR

Five Mistakes That Give Junior IEs Away

✗ Reading only the bar's total height and never looking at what the individual colored segments actually are.

✗ Treating a Combine/Rearrange/Simplify 25% saving as guaranteed — it's a planning estimate, not a promise until it's implemented and re-measured.

✗ Missing an MTM-inferred segment because it wasn't manually tagged — check the tooltip, not just the color.

✗ Recommending a proposed-state number to management without checking which elements it assumes get fixed.

✗ Forgetting this chart's proposed math must match fill_index.py exactly — if your hand-calculation disagrees with the Index sheet, find out why before presenting either one.

FROM PENCIL-AND-PAPER TO THE SOFTWARE

What generate_yamazumi.py Does For You

Every station on your line, built as a true stacked chart — in two formats you can use differently.

1	**Yamazumi_Chart.html**	Interactive, browser-based, with a Present/Proposed toggle — walk a review meeting through both states of every station live.
2	**Yamazumi_Chart.xlsx**	The same data as a native Excel stacked bar chart with a Takt reference line — drop straight into a report or a printed wall chart.
3	**MTM-inferred tagging, automatic**	Every EXCEED element without a manual tag gets Layer 2's fallback logic applied — nothing silently passes as “fine.”
4	**Proposed-state math, locked to fill_index.py**	The exact three rules from Layer 3 — guaranteed to match the Index sheet, every time it's regenerated.

KEY TAKEAWAYS

Recap — Say This Back In Your Own Words

- Yamazumi means “to stack up” — a chart built so imbalance needs no interpretation.
- Every segment's color is a claim: Eliminate, Combine, Rearrange, Simplify, or plain VA — read the legend before the bar.
- An untagged element can still be flagged, through the MTM-inferred fallback — silence isn't the same as “fine.”
- Proposed-state math is exactly three rules — memorize them and you can hand-check any chart in seconds.

TALK IT THROUGH — 3 QUESTIONS FOR YOUR NEXT LINE WALK

1. Pick your line's tallest bar — name its segments by color before you look at the underlying data. Were you right?
2. Find one untagged element on your own sheet — has anyone actually checked whether MTM would flag it EXCEED?
3. Hand-calculate the proposed height of one real station's bar — does it match what the software reports?