Ellingham Diagram Data Extraction

Technical Guide

Honda CALPHAD Project — MSE 4381 Senior Design

February 4, 2026

Contents

1 Introduction

1.1 Objective

Extract Gibbs energies of formation (ΔG_f°) for oxide materials from Thermo-Calc's TCOX14 database to create an Ellingham diagram. This diagram compares oxide stability and answers a key question for the Cu removal project:

Research Question

Can copper thermodynamically reduce ceramic oxides (Al_2O_3, MgO, SiO_2, TiO_2)?

1.2 What is an Ellingham Diagram?

An Ellingham diagram plots ΔG_f° per mole O_2 versus temperature for metal oxide formation reactions.

Key principles:

- Lower (more negative) ΔG = more stable oxide
- A metal can reduce another metal's oxide only if its oxide line is *below* the other
- The vertical gap between lines indicates the thermodynamic driving force

2 Method

2.1 Software and Database

- **Thermo-Calc 2025b** with **TC-Python** API
- **TCOX14** database (comprehensive oxide thermodynamics)
- Run on **OSU ETS Virtual Machine** (Windows, Thermo-Calc license)

2.2 Oxides Calculated

Table 1: Oxide phases and formation reactions normalized per mole O_2

Oxide	Mineral Name	TC Phase	Reaction (per mol O_2)
Cu_2O	Cuprite	CUPRITE#1	$4Cu + O_2 \rightarrow 2Cu_2O$
CuO	Tenorite	CUO#1	$2Cu + O_2 \rightarrow 2CuO$
Al_2O_3	Corundum	CORUNDUM#1	$\frac{4}{3}Al + O_2 \rightarrow \frac{2}{3}Al_2O_3$
MgO	Periclase	HALITE#1	$2Mg + O_2 \rightarrow 2MgO$
SiO_2	Quartz	QUARTZ#1	$Si + O_2 \rightarrow SiO_2$
TiO_2	Rutile	RUTILE#1	$Ti + O_2 \rightarrow TiO_2$
FeO	Wüstite	HALITE#1	$2Fe + O_2 \rightarrow 2FeO$

2.3 Temperature Range

- Range: 500 K to 2000 K
- Step size: 50 K
- Total: 31 data points per oxide

3 Calculation Details

3.1 Formation Energy Formula

For the general reaction: Metal + O_2 → Oxide

$$\Delta G_f^\circ = G_{oxide}^\circ - n \cdot G_{metal}^\circ - G_{O_2}^\circ \tag{1}$$

Where:

- G_{oxide}° = Gibbs energy of the oxide phase (per formula unit)
- G_{metal}° = Gibbs energy of pure metal reference state
- $G_{O_2}^\circ$ = Gibbs energy of O_2 gas at 1 atm
- n = stoichiometric coefficient for metal

3.2 Normalization per Mole O_2

All values are normalized per mole O_2 consumed for direct comparison on the Ellingham diagram. For example:

- **Cu_2O:** $\Delta G_{per\ O_2} = 2 \times G(Cu_2O) - 4 \times G(Cu) - G(O_2)$
- **Al_2O_3:** $\Delta G_{per\ O_2} = \frac{2}{3} \times G(Al_2O_3) - \frac{4}{3} \times G(Al) - G(O_2)$

3.3 TC-Python Implementation

The extraction script performs four steps at each temperature:

1. **O_2 Reference:** Calculate $G(O_2)$ using pure O system
   - TC returns GM per mole O atoms → multiply by 2 for per mole O_2
2. **Metal Reference:** Calculate $G(metal)$ at near-zero oxygen ($X_O = 0.0001$)
3. **Oxide Phase:** Set stoichiometric composition, calculate equilibrium
   - Extract individual phase energy: `GM(CUPRITE#1)`
   - Convert per-atom to per-formula-unit: multiply by atoms per formula
4. **Formation Energy:** Apply the ΔG_f formula

Table 2: Bugs encountered and solutions implemented

Problem	Cause	Solution
Wrong sign for Cu_2O (+97 instead of -191 kJ/mol)	Two-phase equilibrium (CUPRITE + FCC_A1); system GM is mixture energy	Extract individual phase energy using `GM(CUPRITE#1)`
Values too small by factor of 2-5	TC returns GM per mole atoms, not per formula unit	Multiply by atoms_per_formula (e.g., 3 for Cu_2O)
O_2 reference off by factor of 2	GM for pure O is per mole O atoms	Multiply by 2 for per mole O_2

3.4 Technical Challenges Solved

4 How the Pipeline Works

4.1 What the Script Actually Does

The extraction script is not doing anything clever. It is a mechanical loop that calls the same Thermo-Calc API function over and over with different inputs. At each temperature, for each oxide, the script asks Thermo-Calc three questions:

1. What is the Gibbs energy of the **pure metal**? (e.g., pure Cu)
2. What is the Gibbs energy of the **oxide**? (e.g., Cu_2O at $X_O = 1/3$)
3. What is the Gibbs energy of **O_2 gas**?

It then subtracts reactants from products to get ΔG of formation, writes one number per oxide per temperature to the CSV, and moves on. There is no curve fitting, no interpolation, and no thermodynamic modeling on our part. The science is entirely inside Thermo-Calc's TCOX14 database, which contains assessed model parameters fitted to decades of experimental calorimetry, EMF measurements, and phase equilibrium data. Our script is a glorified database query.

The equivalent GUI workflow would be: Property Diagram → select binary system (e.g., Cu-O) → set $X(O) = 0.333$ → plot GM vs T → read off values. Repeat for each oxide. Export to Excel. Make a chart. That takes roughly 45 minutes of clicking for all 7 oxides. The TC-Python script automates the same process in ∼5 minutes of runtime.

4.2 What the Notebook Does with the Data

The Marimo notebook reads the CSV and does three things:

1. Reads the `dG_[oxide]_per_O2` column for each oxide
2. Divides by 1000 to convert J → kJ
3. Calls `plt.plot(T_C, dG_vals)` to plot temperature vs. ΔG

That is the entire analysis. Matplotlib's `plot()` connects each consecutive pair of data points with a straight line segment. At 50 K spacing, the 31 dots are close enough that the result looks like a smooth, continuous curve.

The interactive controls (temperature slider, oxide selector) are cosmetic. The slider marks a vertical line at the selected temperature and the selector filters which oxides to display. Neither changes the underlying data.

Note on the slider: The Marimo slider uses 25°C steps from 500 to 1700°C, giving 49 positions. This controls only the vertical temperature marker, not the data resolution. The actual data is 31 points at 50 K spacing from 500 to 2000 K.

4.3 Resolution: Why 31 Points Is Sufficient

The 50 K step size was chosen for convenience, not necessity. To increase resolution, change one line in `extract_oxide_gibbs.py`:

```
# Line 24 - current setting (31 points, ~5 min runtime)
T_STEP = 50

# Higher resolution (301 points, ~10 min runtime)
T_STEP = 5

# Maximum resolution (1501 points, excessive)
T_STEP = 1
```

The curves would be indistinguishable at any of these resolutions. This is because ΔG vs. T for oxide formation is nearly linear, a consequence of the Gibbs-Helmholtz relation:

$$\Delta G^\circ = \Delta H^\circ - T\Delta S^\circ \tag{2}$$

ΔH° and ΔS° are approximately constant over a given temperature range (they change slowly due to heat capacity). So ΔG vs. T is nearly a straight line. The slight curvature comes from C_p effects and phase transitions (for example, Cu_2O melting at 1235°C produces a visible slope change). 31 points at 50 K spacing captures this shape completely. The interpolation error between our connected line segments and a "true" continuous curve is less than 1 kJ/mol O_2, which is meaningless when the stability gaps we are examining are ∼800 kJ.

4.4 Is This a "Real" Ellingham Diagram?

Yes. The data points are exactly the same numbers Thermo-Calc would produce in the GUI. We query the same TCOX14 database, the same equilibrium solver, and the same thermodynamic models. The only difference is resolution: we computed 31 points, whereas the GUI's Property Diagram calculator would compute several hundred. But as shown above, this does not affect the result.

Every textbook Ellingham diagram is constructed the same way. Ellingham's original 1944 paper was plotted from tabulated data points connected by line segments. There is no closed-form analytical function that produces a continuous Ellingham curve. Our version uses a commercial thermodynamic database instead of JANAF tables, but the method is identical.

4.5 CSV Output Format

The output CSV contains 31 rows (one per temperature) and 44 columns. For each oxide, the following columns are recorded (shown here using Cu_2O as an example):

Table 3: CSV column descriptions (repeated for each of 7 oxides)

Column	Description	Example (1000 K)
`T_K`	Temperature in Kelvin	1000
`T_C`	Temperature in Celsius	726.85
`GM_Cu2O`	Gibbs energy of Cu_2O phase, per mole atoms (J)	−99,489
`GM_system_Cu2O`	Total system energy at that composition (J)	−99,436
`G_metal_Cu2O`	Gibbs energy of pure Cu reference (J)	−46,339
`dG_Cu2O_per_O2`	**Final result:** ΔG_f° in J/mol O_2	−190,813
`phases_Cu2O`	Stable phases found by TC	CUPRITE#1;FCC_A1#1
`oxide_phase_Cu2O`	Phase used for the calculation	CUPRITE#1

This pattern repeats for all 7 oxides (Cu_2O, CuO, Al_2O_3, MgO, SiO_2, TiO_2, FeO). The only column the Marimo notebook reads is `dG_[oxide]_per_O2`. Everything else is retained for traceability so the calculation can be verified at any step.

5 Results

5.1 Gibbs Energies at 1000 K

Table 4: Gibbs energy of formation per mole O_2 at 1000 K (727°C)

Oxide	ΔG_f° (kJ/mol O_2)	Phase	Stability
MgO	−986.8	HALITE#1	Most stable
Al_2O_3	−907.5	CORUNDUM#1	
TiO_2	−760.2	RUTILE#1	
SiO_2	−730.2	QUARTZ#1	
FeO	−411.2	HALITE#1	
Cu_2O	−190.8	CUPRITE#1	
CuO	−132.0	CUO#1	Least stable

5.2 Key Finding

Thermodynamic Conclusion

Cu_2O is the LEAST stable oxide among all candidates, with ΔG_f° approximately 800 kJ/mol O_2 less negative than MgO or Al_2O_3.
This means copper **cannot** reduce these ceramics:

$$Cu + Al_2O_3 \nrightarrow Cu_2O + Al \quad (\Delta G_{rxn} \gg 0)$$

The thermodynamic driving force is approximately +800 kJ/mol O_2 in the *wrong* direction.

5.3 Physical Stability: Melting Points

Cu oxides also melt at lower temperatures than the ceramic candidates:

Table 5: Melting points and phase behavior at steelmaking temperatures

Material	Melting Point (°C)	Status at 1500°C
Cu_2O	1235	**Liquid**
CuO	1326	**Liquid**
FeO	1377	**Liquid**
SiO_2	1713	Solid
TiO_2	1843	Solid
Al_2O_3	2072	Solid
MgO	2852	Solid

6 Implications for the Project

6.1 What This Tells Us

1. **Direct oxide reduction is thermodynamically impossible** — Cu cannot "steal" oxygen from Al_2O_3, MgO, SiO_2, or TiO_2
2. **Previous Cu capture observations** must involve different mechanisms

6.2 Alternative Cu Removal Mechanisms

6.3 Recommended Next Steps

1. Model Cu solubility in Al_2O_3 and MgO using Thermo-Calc
2. Calculate $CuAl_2O_4$ spinel stability conditions
3. Compare predictions with last year's senior design experimental results
4. Design experiments to test the dominant mechanism

Table 6: Possible mechanisms for Cu capture by ceramics

Mechanism	Description	Key Factor
Solid Solution	Cu dissolves into oxide lattice (substitutional or interstitial)	High temperature (entropy-driven)
Spinel Formation	$CuO + Al_2O_3 \rightarrow CuAl_2O_4$	Requires Cu oxidation first
Surface Adsorption	Cu atoms adsorb on ceramic particle surfaces	Surface area, porosity
Physical Infiltration	Fe-Cu melt infiltrates porous ceramic (not Cu-selective)	Porosity, capillary pressure

7 Step-by-Step: Running on OSU Lab Machines

This section documents exactly how the extraction was performed on the OSU ETS Virtual Machine.

7.1 Accessing the Virtual Machine

1. Go to `https://ets.engineering.osu.edu/`
2. Log in with OSU credentials
3. Select a Windows VM with Thermo-Calc installed
4. Open **File Explorer** and navigate to your U: drive

7.2 Getting the Code from GitHub

Since Git is not installed on the lab VMs, download the repository as a ZIP file:

1. Open a browser and go to: `https://github.com/dimascad/honda-calphad`
2. Click the green **"Code"** button
3. Select **"Download ZIP"**
4. Extract the ZIP to your U: drive, e.g., `U:\4381\honda-calphad\`

7.3 Opening the Command Prompt

1. Press `Win + R`, type `cmd`, press Enter
2. Navigate to the script directory:

```
cd U:\4381\honda-calphad\simulations\tcpython
```

7.4 Running the TC-Python Script

Thermo-Calc includes its own Python distribution. Use this exact path:

```
"C:\Program Files\Thermo-Calc\2025b\python\python.exe" extract_oxide_gibbs.py
```

Important: The quotes around the path are required because of the spaces in "Program Files".

7.5 Expected Output

The script takes approximately 4–5 minutes to complete. You will see:

```
======================================================================
TC-Python: Oxide Formation Energies for Ellingham Diagram
======================================================================
Temperature range: 500-2000 K (31 points)

Connected to Thermo-Calc

--- Getting O2 reference energies ---
  O2 reference at 1000K: -220766.0 J/mol-O2

--- Processing Cu2O ---
  Getting CU reference...
  Getting Cu2O oxide phase...
  Completed: 31/31 temperatures
  dG at 1000K: -190.8 kJ/mol O2
  Stable phases: CUPRITE#1;FCC_A1#1
  Used for calc: CUPRITE#1

[... continues for each oxide ...]

======================================================================
Writing to ...\data\tcpython\raw\oxide_gibbs_energies.csv
Done! 31 rows written.
======================================================================
```

7.6 Retrieving the Output

The CSV file is saved to:

```
U:\4381\honda-calphad\data\tcpython\raw\oxide_gibbs_energies.csv
```

To get the file back to your local machine:

- **Option A:** Copy to OneDrive/Google Drive from the VM
- **Option B:** Email the CSV to yourself
- **Option C:** Download directly from the VM browser

Table 7: Common issues and solutions

Issue	Solution
"python is not recognized"	Use the full path with quotes: `"C:\Program Files\Thermo-Calc\2025b\python\python.exe"`
"tc_python not found"	You must use Thermo-Calc's bundled Python, not system Python
"TCOX14 not found"	Check available databases with `check_databases.py` first
IONIC_LIQ warnings	These are normal and don't affect results

Table 8: Output files from this analysis

File	Description
`data/tcpython/raw/oxide_gibbs_energies.csv`	Raw data (31 temps × 7 oxides)
`data/tcpython/ellingham_diagram_tcox14.png`	Static plot (PNG)
`data/tcpython/ellingham_diagram_tcox14.pdf`	Vector plot (PDF)
`simulations/notebooks/ellingham_diagram.py`	Interactive Marimo notebook
`simulations/tcpython/extract_oxide_gibbs.py`	TC-Python extraction script

7.7 Troubleshooting

8 Files and Reproduction

8.1 Files Generated

8.2 How to Reproduce

On OSU ETS Virtual Machine:

```
cd U:\4381\honda-calphad\simulations\tcpython
"C:\Program Files\Thermo-Calc\2025b\python\python.exe" extract_oxide_gibbs.py
```

Output: `data/tcpython/raw/oxide_gibbs_energies.csv`

8.3 Interactive Notebook

The Marimo notebook can be accessed via GitHub import:

`https://github.com/dimascad/honda-calphad/blob/main/simulations/notebooks/ellingham_diagram.py`

9 References

1. Thermo-Calc Software, *TCOX14 Database Documentation*, 2024.
2. Ellingham, H.J.T. (1944). "Reducibility of oxides and sulphides in metallurgical processes." *J. Soc. Chem. Ind.* 63: 125–133.

3. Gaskell, D.R. *Introduction to the Thermodynamics of Materials*, 5th ed. Taylor & Francis, 2008.

4. NIST-JANAF Thermochemical Tables, `https://janaf.nist.gov/`

Document generated: February 4, 2026