EPISTEMEUS
ESSAYS

Volume 1 ; Issue 2 01.08.2026

Rangkaian Latch

Hosea Pratama Sinaga

1 Hosea Pratama Sinaga
EPISTEMEUS E S SAYS
Volume 1 ; Issue 1
Rangkaian Latch
Hosea Pratama Sinaga

ABSTRAK

omputer membutuhkan rangkaian yang mampu mempertahankan data K meskipun masukan sudah berubah. Salah satu elemen penyimpanan paling sederhana dalam elektronika digital adalah latch. Esai ini membahas prinsip kerja rangkaian latch melalui rangkaian sekuensial, umpan balik, serta fungsi masukan Set, Reset, Enable, dan Data. SR Latch digunakan untuk menjelaskan keadaan set, reset, hold, dan kondisi invalid ketika masukan Set dan Reset aktif bersamaan. D Latch memakai satu jalur data agar kombinasi masukan lebih mudah dikendalikan. Ketika Enable bernilai 1, keluaran mengikuti nilai masukan. Ketika Enable bernilai 0, keluaran mempertahankan keadaan terakhir yang telah tersimpan. Simulasi rangkaian membantu siswa mengamati perubahan keluaran, menguji tabel kebenaran, dan memahami cara satu bit informasi disimpan. Pembahasan latch dapat menjadi dasar untuk mempelajari flip-flop, register, pencacah, dan memori digital. Materi ini diarahkan untuk memperkenalkan konsep penyimpanan data melalui rangkaian yang sederhana, dapat diamati, dan mudah diuji oleh pelajar. Kata kunci: rangkaian latch, rangkaian sekuensial, elemen memori, SR Latch, D Latch, elektronika digital. Pendahuluan Telepon genggam tetap mengingat tingkat kecerahan layar yang dipilih, mesin penghitung mempertahankan angka sebelum operasi berikutnya dimasukkan, dan komputer menyimpan keadaan program walaupun jutaan sinyal terus berubah di dalam rangkaiannya. Kemampuan tersebut berawal dari satu persoalan kecil yang jarang terlihat oleh pengguna, yaitu cara menyimpan keadaan 0 atau 1. Bagaimana sebuah rangka-

ian dapat mengingat nilai lama ketika masukan yang diterimanya sudah berubah? Gerbang logika biasa Hosea Pratama Sinaga 2 menghasilkan keluaran berdasarkan masukan yang sedang diberikan. Sistem digital membutuhkan bagian lain yang dapat mempertahankan informasi agar hasil sebelumnya masih tersedia untuk langkah berikutnya. Bagian penyimpanan terkecil ini menjadi dasar bagi register, pencacah, memori, dan rangkaian pengendali. Latch memberi contoh paling sederhana karena rangkaiannya dapat diamati dari hubungan beberapa gerbang, aliran umpan balik, dan perubahan dua keluaran. Dari satu bit yang tersimpan, siswa dapat mulai memahami bagaimana komputer membangun kemampuan mengingat dalam ukuran yang jauh lebih besar (Harris & Harris, 2012; Nisan & Schocken, 2021). Rangkaian digital dapat dibedakan berdasarkan cara keluarannya dibentuk. Rangkaian kombinasional membaca masukan saat ini, memprosesnya, lalu menghasilkan keluaran sesuai susunan gerbang yang digunakan. Saat nilai masukan berubah, keluaran mengikuti aturan yang telah dirancang. Rangkaian sekuensial bekerja memakai masukan saat ini bersama keadaan yang tersimpan dari proses sebelumnya. Mengapa keluaran masa lalu perlu dimasukkan kembali ke dalam rangkaian? Nilai tersebut berfungsi sebagai jejak yang memberi informasi mengenai keadaan terakhir sistem. Jalur yang membawa sebagian keluaran kembali menuju masukan disebut umpan balik atau feedback. Umpan balik membuat rangkaian mempunyai riwayat, sehingga dua kondisi masukan yang sama dapat menghasilkan respons berbeda jika keadaan awalnya tidak sama. Materi sumber menempatkan elemen memori sebagai bagian yang membedakan rangkaian sekuensial dari rangkaian kombinasional serta menjelaskan bahwa keluaran sebelumnya digunakan kembali pada proses berikutnya (Modul Rangkaian Latch, n.d.; Mano & Ciletti, 2018). Latch merupakan elemen memori yang mampu mempertahankan satu bit informasi. Nilai yang tersimpan ditampilkan pada keluaran Q, sedangkan keluaran pasangannya biasa ditulis sebagai $\bar{Q}$. Kedua keluaran umumnya memiliki nilai berlawanan saat rangkaian berada pada keadaan yang benar. Hubungan silang antargerbang membuat keluaran pertama memengaruhi gerbang kedua, lalu keluaran kedua kembali memengaruhi gerbang pertama. Susunan tersebut membentuk dua keadaan stabil yang dapat dipertahankan selama rangkaian memperoleh daya. Kapan sebuah latch menyimpan nilai 1 dan kapan nilai itu dihapus menjadi 0? Pertanyaan tersebut dijawab lewat SR Latch, yaitu latch dengan masukan Set dan Reset. Set dipakai untuk menempatkan Q pada nilai 1, Reset menempatkan Q pada nilai 0, sedangkan keadaan tertentu mempertahankan nilai lama atau hold. Kombinasi masukan lain dapat menghasilkan keadaan terlarang karena kedua keluaran kehilangan hubungan komplemennya. Pembacaan SR Latch mengajarkan bahwa penyimpanan digital muncul dari susunan gerbang yang saling menjaga keadaan, bukan dari ruang penyimpanan berbentuk kotak seperti yang tampak pada antar-

muka komputer (Wakerly, 2019; Roth & Kinney, 2014). 3 Hosea Pratama Sinaga Pengendalian latch menjadi lebih teratur saat rangkaian dilengkapi masukan Enable. Enable bertindak sebagai izin yang menentukan apakah data baru boleh memengaruhi keluaran. Pada D Latch, satu masukan data bernama D menggantikan pasangan Set dan Reset yang harus diatur secara terpisah. Ketika Enable aktif, nilai D diteruskan menuju Q. Saat Enable tidak aktif, perubahan D tidak mengubah nilai yang sudah tersimpan. Apa yang terjadi jika D berubah beberapa kali selama Enable masih aktif? Keluaran dapat mengikuti perubahan tersebut karena latch bersifat peka terhadap tingkat sinyal atau level-sensitive. Cara kerja ini berbeda dari flip- flop yang umumnya menerima data pada peralihan tertentu dari sinyal pemicu, misalnya saat sinyal berubah dari 0 ke 1. Perbedaan tersebut menjelaskan mengapa waktu perubahan masukan perlu diperhatikan dalam perancangan sistem digital. D Latch juga menghindari kombinasi Set dan Reset aktif bersamaan karena rangkaiannya membentuk kedua jalur tersebut dari satu nilai D dan nilai kebalikannya (Tocci et al., 2017; Even & Medina, 2012). Simbol gerbang dan tabel keadaan sering terlihat rumit saat siswa belum pernah mengubah masukannya sendiri. Simulator dapat memberi bentuk yang lebih mudah diamati karena setiap sakelar dapat ditekan, jalur sinyal dapat ditelusuri, dan keluaran Q dapat dilihat setelah Set, Reset, Enable, atau D diubah. Bagaimana siswa mengetahui bahwa keluaran benar-benar menyimpan nilai lama, bukan sekadar terlambat berubah? Mereka dapat menonaktifkan Enable, mengganti nilai D beberapa kali, lalu memeriksa apakah Q tetap bertahan. Percobaan serupa dapat dilakukan pada SR Latch untuk membedakan keadaan set, reset, hold, dan kondisi terlarang. Prediksi ditulis sebelum simulasi dijalankan, hasilnya dicatat, lalu perbedaan antara dugaan dan keluaran rangkaian dibahas. Simulasi memberi ruang untuk mengulang pengujian tanpa mengganti komponen fisik setiap kali terjadi kesalahan. Penelitian mengenai pembelajaran berbasis simulasi mencatat bahwa eksplorasi menjadi lebih terarah saat peserta didik memperoleh pertanyaan penuntun, tujuan pengamatan, dan kesempatan menjelaskan hasil percobaannya sendiri (de Jong & van Joolingen, 1998; Rutten et al., 2012). Esai berjudul Rangkaian Latch membahas langkah awal yang mengubah gerbang logika menjadi elemen penyimpanan. Pembahasan diarahkan pada perbedaan rangkaian kombinasional dan sekuensial, fungsi umpan balik, prinsip SR Latch, peran Enable, kondisi hold, keadaan terlarang, serta kerja D Latch. Di mana konsep sederhana ini muncul setelah siswa selesai mempelajari tabel kebenarannya? Latch menjadi bagian awal dalam pengembangan flip-flop, register, pencacah, mesin keadaan, dan berbagai sistem memori digital. Materinya dapat dikenalkan kepada pelajar lewat analogi tombol simpan, sakelar, lampu indikator, permainan keadaan, serta simulasi rangkaian. Urutan tersebut membawa siswa dari gejala yang terlihat menuju alasan teknis di balik kemampuan komputer mempertahankan

data. PembelajarHosea Pratama Sinaga 4 an ilmu komputer tingkat sekolah juga memasukkan perangkat komputasi, representasi data, dan cara sistem menyimpan serta mengolah informasi sebagai dasar untuk membaca teknologi secara terstruktur. Satu latch memang hanya menyimpan satu bit, tetapi satu bit itu cukup untuk membuka pembahasan mengenai cara mesin mengenali masa lalu dan menentukan keadaan berikutnya (Nahin, 2017; Computer Science Teachers Association, 2017).

Teori Dasar

a. Rangkaian Digital dan Kebutuhan Menyimpan Data Rangkaian digital bekerja memakai dua keadaan logika yang ditulis sebagai 0 dan 1. Kedua simbol tersebut dapat mewakili banyak hal, seperti sakelar mati dan menyala, jawaban salah dan benar, atau tegangan rendah dan tinggi. Satu keadaan biner disebut bit. Komputer menggabungkan banyak bit untuk membentuk angka, huruf, gambar, suara, serta instruksi program. Proses pengolahan data belum cukup jika rangkaian tidak mampu menyimpan hasil yang telah diperoleh. Bayangkan kalkulator yang langsung melupakan angka pertama sesaat setelah tombol operasi ditekan. Perhitungan berikutnya tidak dapat dilanjutkan karena tidak ada nilai yang dipertahankan. Kebutuhan serupa muncul pada komputer, mesin otomatis, penghitung digital, dan sistem kendali. Bagaimana rangkaian yang hanya menerima tegangan dapat mengingat sebuah nilai? Penyimpanan dasar dibentuk dengan menghubungkan keluaran kembali menuju bagian masukan. Hubungan tersebut membuat keadaan lama tetap memengaruhi proses yang sedang berlangsung. Latch menjadi salah satu bentuk paling sederhana dari elemen penyimpanan tersebut. Satu latch umumnya menyimpan satu bit yang nilainya dapat berupa 0 atau 1 selama catu daya masih tersedia (Harris & Harris, 2012; Mano & Ciletti, 2018).

b. Rangkaian Kombinasional dan Rangkaian Sekuensial Rangkaian kombinasional menghasilkan keluaran berdasarkan nilai masukan yang sedang diterimanya. Gerbang AND, OR, XOR, decoder, multiplexer, dan adder termasuk dalam kelompok ini. Jika masukan berubah, keluaran ikut berubah sesuai aturan logikanya. Rangkaian tersebut tidak memiliki informasi mengenai apa yang terjadi sebelumnya. Sebuah gerbang AND yang menerima masukan 1 dan 1 selalu menghasilkan 1, tanpa memedulikan keadaan beberapa detik sebelumnya. Rangkaian sekuensial memakai cara kerja yang berbeda karena keluarannya dipengaruhi oleh masukan sekarang dan keadaan yang telah tersimpan. Mengapa keadaan sebelumnya masih diperlukan? Beberapa sistem harus mengetahui urutan kejadian, bukan sekadar membaca satu kondisi. Mesin penghitung perlu mengingat angka terakhir sebelum menaikkan hitungannya. Sistem lampu lalu lintas perlu mengetahui 5 Hosea Pratama Sinaga

fase yang sedang aktif sebelum berpindah ke fase selanjutnya. Rangkaian sekuensial memperoleh kemampuan tersebut dari elemen memori. Ma-

teri sumber menggambarkan keluaran sebelumnya yang digunakan kembali sebagai bagian dari masukan pada proses berikutnya. Susunan itu membuat rangkaian mempunyai keadaan internal yang dapat bertahan saat masukan luar tidak berubah. Perbedaan ini menjadi dasar untuk memahami mengapa latch tidak dapat dibaca hanya seperti gerbang logika biasa (Wakerly, 2006; Roth & Kinney, 2014). Jenis rangkaian Dasar pembentukan keluaran Memiliki memori Kombinasional Masukan saat ini Tidak Sekuensial Masukan saat ini dan keadaan sebelumnya Ya

c. Umpan Balik dan Keadaan Stabil Umpan balik atau feedback adalah hubungan yang membawa sebagian keluaran kembali menuju masukan rangkaian. Pada sistem suara, umpan balik yang tidak terkendali dapat menghasilkan bunyi mendenging. Pada latch, umpan balik sengaja dirancang agar dua gerbang saling mempertahankan keadaan. Keluaran gerbang pertama masuk ke gerbang kedua, lalu keluaran gerbang kedua kembali menuju gerbang pertama. Hubungan silang tersebut menghasilkan rangkaian yang memiliki dua keadaan stabil. Keadaan pertama menyimpan Q = 1, sedangkan keadaan kedua menyimpan Q = 0. Mengapa keluaran tidak kembali ke keadaan awal setelah tombol masukan dilepas? Nilai yang tersimpan terus diperkuat melalui jalur umpan balik. Saat Q bernilai 1, nilai tersebut membantu mempertahankan keluaran pasangannya pada 0. Keluaran pasangan itu kembali menjaga Q agar tetap berada pada 1. Proses yang sama berlaku secara terbalik ketika Q menyimpan 0. Dua keadaan stabil membuat latch disebut sebagai rangkaian bistabil. Istilah bi berarti dua, sedangkan stabil berarti keadaan tersebut dapat dipertahankan. Perubahan hanya terjadi ketika masukan tertentu memaksa rangkaian berpindah dari satu keadaan menuju keadaan lain. Prinsip bistabil inilah yang memungkinkan sepasang gerbang menyimpan satu bit informasi tanpa memakai komponen penyimpanan berbentuk ruang atau kotak fisik (Floyd, 2015; Tocci et al., 2017).

d. Perbedaan Latch dan Flip-Flop Latch dan flip-flop sama-sama digunakan sebagai elemen memori, tetapi keduanya merespons sinyal kendali dengan cara berbeda. Latch bersifat peka terhadap tingkat sinyal atau level-sensitive. Selama masukan Enable berada pada tingkat aktif, data masih dapat memengaruhi keluaran. Jika Enable bernilai 1 selama beberapa saat, perubahan data yang terjadi dalam rentang tersebut dapat diteruskan menuju Q. Flip-flop umumnya bersifat peka terhadap tepi sinyal atau edge-sensitive. Data dibaca hanya saat terjadi perubahan Hosea Pratama Sinaga 6

tertentu pada clock, misalnya dari 0 menuju 1. Apa perbedaan paling mudah untuk dibayangkan? Latch dapat diibaratkan sebagai pintu yang terbuka selama Enable aktif. Data dapat lewat selama pintu itu belum ditutup. Flip-flop lebih mirip kamera yang mengambil satu gambar tepat saat tombol ditekan. Perubahan sebelum dan sesudah momen tersebut tidak langsung mengubah data yang telah ditangkap. Materi sumber membedakan latch yang bekerja berdasarkan tingkat Enable dari flip-flop yang dipicu oleh pe-

rubahan sinyal. Pemilihan latch atau flip-flop bergantung pada kebutuhan waktu, susunan rangkaian, dan cara data dipindahkan. Pembahasan awal mengenai latch membantu siswa memahami konsep penyimpanan sebelum masuk ke sistem clock yang lebih teratur (Brown & Vranesic, 2014; Katz & Borriello, 2005).

e. SR Latch SR Latch merupakan latch yang mempunyai dua masukan utama bernama Set dan Reset. Huruf S berasal dari kata Set, sedangkan R berasal dari kata Reset. Set digunakan untuk menyimpan keadaan Q = 1. Reset digunakan untuk mengubah Q menjadi 0. Rangkaian juga mempunyai dua keluaran yang disebut Q dan $\bar{Q}$. Simbol $\bar{Q}$ dibaca Q bar atau komplemen Q. Dalam keadaan normal, kedua keluaran memiliki nilai yang berlawanan. Jika Q bernilai 1, $\bar{Q}$ bernilai 0. Jika Q bernilai 0, $\bar{Q}$ bernilai 1. Mengapa dibutuhkan dua keluaran? Beberapa rangkaian memerlukan nilai asli dan nilai kebalikannya pada waktu yang sama. Dua keluaran tersebut juga membantu memperlihatkan apakah latch sedang bekerja dalam keadaan normal. SR Latch pada materi dilengkapi masukan Enable atau E. Set dan Reset baru dapat memengaruhi keluaran saat Enable bernilai 1. Ketika Enable bernilai 0, jalur masukan tidak meneruskan perubahan menuju bagian penyimpanan. Materi menampilkan SR Latch dengan tiga masukan, yaitu Set, Reset, dan Enable, serta dua keluaran yang dipengaruhi oleh masukan ketika Enable diaktifkan. Rangkaian ini menjadi contoh paling langsung untuk melihat proses menulis, menghapus, dan mempertahankan satu bit (Givone, 2003; Holdsworth & Woods, 2002).

f. Kondisi Set, Reset, Hold, dan Invalid Saat Enable aktif, SR Latch memiliki empat kombinasi masukan Set dan Reset. Kombinasi S = 1 dan R = 0 menghasilkan keadaan set, sehingga Q menjadi 1. Kombinasi S = 0 dan R = 1 menghasilkan keadaan reset, sehingga Q menjadi 0. Ketika S dan R sama-sama 0, tidak ada perintah untuk mengganti data. Q mempertahankan keadaan sebelumnya. Keadaan tersebut disebut hold, latch, atau memory. Jika Q sebelumnya bernilai 1, nilainya tetap 1. Jika Q sebelumnya bernilai 0, nilainya tetap 0. Apa yang terjadi saat S dan R sama-sama bernilai 1? Pada SR Latch berbasis gerbang NOR, kedua keluaran dapat dipaksa menjadi 0. Kondisi ini merusak hubungan normal bahwa Q dan $\bar{Q}$ harus saling berlawanan. Saat kedua masukan dikembalikan ke 7 Hosea Pratama Sinaga

0 hampir bersamaan, keadaan akhir dapat bergantung pada gerbang mana yang merespons lebih dahulu. Selisih waktu yang sangat kecil dapat membuat hasilnya sulit diprediksi. Kombinasi tersebut disebut invalid atau terlarang. Materi juga menyebut bahwa keluaran berikutnya tidak dapat dipastikan ketika Set dan Reset sama-sama aktif. Kondisi invalid menjadi alasan utama lahirnya rancangan latch yang hanya menerima satu masukan data (Hayes, 1993; Lala, 2007). Enable S R Q berikutnya Keadaan 0 X X Q sebelumnya Hold 1 0 0 Q sebelumnya Hold 1 0 1 0 Reset 1 1 0 1 Set 1 1 1

Tidak pasti Invalid

Huruf X berarti nilai masukan dapat berupa 0 atau 1 karena tidak memengaruhi keluaran pada kondisi tersebut.

g. D Latch D Latch dikembangkan untuk menghindari kemungkinan Set dan Reset aktif bersamaan. Huruf D biasanya dibaca sebagai Data. Rangkaian ini mempunyai satu masukan data D dan satu masukan kendali Enable. Bagian dalamnya tetap dapat dibentuk dari SR Latch, tetapi jalur Set dan Reset dibuat dari nilai yang berlawanan. Saat D bernilai 1, jalur Set menerima keadaan aktif dan jalur Reset menerima keadaan tidak aktif. Saat D bernilai 0, jalur Reset aktif dan jalur Set tidak aktif. Sebuah gerbang NOT dipakai untuk menghasilkan kebalikan dari D. Mengapa susunan ini menghilangkan kondisi invalid? Set dan Reset tidak dapat bernilai 1 secara bersamaan karena salah satunya selalu menerima nilai D, sedangkan yang lain menerima nilai kebalikannya. Ketika Enable bernilai 1, Q mengikuti nilai D. Jika D berubah menjadi 1, Q menjadi 1. Jika D berubah menjadi 0, Q menjadi 0. Ketika Enable bernilai 0, Q mempertahankan data terakhir yang telah diterima. Materi sumber menjelaskan bahwa D Latch digunakan untuk mengatasi kondisi terlarang pada SR Latch dan mempertahankan keluaran ketika Enable tidak aktif. Satu masukan data membuat D Latch lebih mudah digunakan sebagai elemen penyimpanan pada sistem digital yang lebih besar (Kleitz, 2012; Nisan & Schocken, 2005). Enable D Q berikutnya Keadaan 0 X Q sebelumnya Menyimpan 1 0 0 Menulis 0 1 1 1 Menulis 1

h. Fungsi Enable dan Sifat Transparan Enable menentukan kapan latch boleh menerima data baru. Saat Enable bernilai 0, jalur menuju bagian penyimpanan ditutup. Perubahan pada D, S, Hosea Pratama Sinaga 8

atau R tidak mengubah Q. Saat Enable bernilai 1, jalur tersebut terbuka dan masukan dapat mengendalikan keadaan keluaran. Pada D Latch, keadaan ketika Enable aktif sering disebut transparan. Istilah transparan berarti nilai D dapat terlihat pada Q selama jalur data masih terbuka. Jika D berubah dari 0 menjadi 1 saat Enable tetap aktif, Q ikut berubah menjadi 1 setelah melewati waktu tunda rangkaian. Jika D kembali menjadi 0, Q juga kembali menjadi 0. Kapan sebuah nilai benar-benar tersimpan? Nilai terakhir pada D dipertahankan ketika Enable berubah menjadi tidak aktif. Sesudah titik itu, D dapat berganti tanpa mengubah Q. Cara kerja ini membuat waktu pengaktifan Enable perlu diperhatikan. Data yang belum stabil saat Enable ditutup dapat menghasilkan keadaan yang tidak diinginkan. Persamaan keadaan D Latch dapat ditulis sebagai

Q berikutnya = E · D + E · Q sebelumnya

Bagian E · D menjelaskan bahwa data diteruskan saat Enable aktif. Bagian E · Q sebelumnya menjelaskan bahwa keadaan lama dipertahankan saat Enable tidak aktif. Persamaan tersebut merangkum kerja D Latch tanpa harus memeriksa seluruh gerbang satu per satu (Mano & Kime, 2008; Balabanian & Carlson, 2001).

i. Keadaan Sekarang dan Keadaan Berikutnya Pembahasan rangkaian sekuensial memakai dua istilah waktu, yaitu keadaan sekarang dan keadaan berikutnya. Keadaan sekarang adalah nilai Q yang tersimpan sebelum masukan baru diproses. Keadaan berikutnya adalah nilai Q setelah rangkaian merespons masukan. Perbedaan ini diperlukan karena kombinasi masukan yang sama belum tentu menghasilkan keluaran yang sama jika keadaan awalnya berbeda. Pada SR Latch, masukan S = 0 dan R = 0 tidak menentukan angka baru. Rangkaian mempertahankan Q yang sudah ada. Jika keadaan sekarang 1, keadaan berikutnya tetap 1. Jika keadaan sekarang 0, keadaan berikutnya tetap 0. Bagaimana tabel dapat menulis dua hasil untuk masukan yang sama? Kolom keadaan sekarang ditambahkan agar pembaca mengetahui nilai yang sedang disimpan. Pada D Latch dengan Enable aktif, keadaan sekarang tidak menentukan hasil karena Q berikutnya langsung mengikuti D. Saat Enable tidak aktif, nilai D tidak menentukan hasil karena Q berikutnya sama dengan Q sekarang. Cara membaca keadaan seperti ini menjadi dasar untuk memahami tabel transisi, diagram keadaan, register, counter, dan finite state machine. Siswa mulai melihat bahwa waktu serta urutan kejadian ikut menentukan perilaku rangkaian, bukan hanya kombinasi masukan yang tampak pada satu saat. 9 Hosea Pratama Sinaga

Enable D Q sekarang Q berikutnya

0 0 0 0
0 1 0 0
0 0 1 1
0 1 1 1
1 0 X 0
1 1 X 1

Pemahaman tabel keadaan membantu siswa membaca latch sebagai elemen yang memiliki riwayat, bukan sebagai gerbang yang langsung melupakan hasil sebelumnya (Comer, 1995; Nelson et al., 1995).

j. Simulasi dan Pembuktian Tabel Kebenaran Simulasi memungkinkan siswa membangun latch tanpa langsung berhadapan dengan kabel, tegangan, atau kesalahan pemasangan komponen fisik. Pengujian SR Latch dapat dimulai dengan mengaktifkan Enable, lalu mencoba kombinasi Set dan Reset secara berurutan. Siswa mencatat Q dan $\bar{Q}$ pada keadaan set, reset, hold, dan invalid. Pengujian D Latch dilakukan dengan mengubah D saat Enable aktif, kemudian menonaktifkan Enable dan mengubah D kembali. Jika Q tetap mempertahankan nilai terakhir, fungsi penyimpanan telah terlihat. Bagaimana cara membuktikan bahwa tabel kebenaran benar-benar sesuai dengan rangkaian? Setiap baris tabel harus diuji satu per satu, bukan hanya memilih kombinasi yang memberi hasil mudah. Keadaan hold juga perlu diuji dua kali dengan keadaan awal berbeda. Q terlebih dahulu dibuat 1, lalu masukan hold diberikan. Pengujian diulang setelah Q dibuat 0. Dua percobaan tersebut membuktikan bahwa hold mempertahankan data lama

dan tidak selalu menghasilkan angka tertentu. Simulasi akan lebih berguna ketika siswa menuliskan prediksi sebelum menekan sakelar. Perbedaan antara prediksi dan hasil dapat diarahkan untuk mencari kesalahan pada pemahaman tabel, arah keluaran, atau fungsi Enable. Materi sumber juga meminta pembuatan SR Latch dan D Latch pada simulator serta pembuktian tabel keadaannya. Eksperimen yang disertai pertanyaan penuntun membantu siswa memahami hubungan sebab- akibat di dalam rangkaian, bukan sekadar meniru susunan gerbang yang sudah jadi (de Jong & van Joolingen, 1998; Rutten et al., 2012).

Teori Tambahan

a. Latch sebagai Sel Memori Satu Bit Satu latch menyimpan satu bit, yaitu satu keadaan 0 atau 1. Kapasitas tersebut terlihat kecil jika dibandingkan dengan memori komputer yang mampu menyimpan miliaran data, tetapi seluruh penyimpanan digital tetap dibangun dari unit-unit kecil yang bekerja bersama. Satu bit dapat dipakai untuk mencatat apakah perangkat sedang aktif, apakah tombol sudah ditekan, apakah Hosea Pratama Sinaga 10

perintah sudah selesai, atau apakah suatu kondisi telah terpenuhi. Bagaimana satu rangkaian mengetahui bahwa nilai yang tersimpan harus tetap dipertahankan? Jalur umpan balik menjaga keluaran lama tetap berada di dalam rangkaian sampai muncul perintah baru. Pada SR Latch, perintah tersebut berasal dari Set dan Reset. Pada D Latch, nilai baru diterima dari masukan D ketika Enable aktif. Prinsip penyimpanan ini menjadi dasar untuk memahami komponen digital yang memiliki keadaan internal. Latch tidak menyimpan angka besar secara langsung. Setiap latch hanya menjaga satu keputusan kecil, lalu banyak latch digabungkan untuk membentuk data yang lebih luas. Susunan tersebut menyerupai deretan kotak yang masing-masing hanya dapat menyimpan satu dari dua keadaan. Empat latch dapat menyimpan pola 1010, delapan latch dapat menyimpan satu byte, dan jumlah yang lebih besar dapat membentuk unit penyimpanan sesuai rancangan sistem (Harris & Harris, 2012; Mano & Ciletti, 2018).

b. Hubungan Latch dengan Register Register adalah kumpulan elemen memori yang digunakan untuk menyimpan beberapa bit secara bersamaan. Jika satu latch menyimpan satu bit, empat latch dapat disusun sebagai register 4 bit. Setiap latch menangani satu posisi data, mulai dari bit paling rendah sampai bit paling tinggi. Saat data 1101 dimasukkan, latch pertama menyimpan 1, latch kedua menyimpan 0, latch ketiga menyimpan 1, dan latch keempat menyimpan 1. Mengapa sistem digital membutuhkan register jika data sudah dapat disimpan di memori utama? Register ditempatkan dekat dengan bagian pengolah agar data yang sedang digunakan dapat diakses dengan cepat. Nilai masukan, hasil sementara, alamat, dan keadaan operasi dapat ditahan di dalam register sebelum dikirim ke bagian lain. D Latch dapat dipakai sebagai penyusun register karena satu jalur

data lebih mudah dikendalikan dibandingkan pasangan Set dan Reset. Enable bersama dapat diberikan kepada seluruh latch agar beberapa bit diterima dalam waktu yang sama. Saat Enable aktif, setiap keluaran mengikuti bit masukannya. Ketika Enable ditutup, seluruh pola data dipertahankan. Hubungan ini membawa siswa dari konsep menyimpan satu lampu menuju penyimpanan satu kelompok angka biner (Wakerly, 2006; Roth & Kinney, 2014).

c. Hubungan Latch dengan Pencacah Digital Pencacah atau counter adalah rangkaian sekuensial yang berpindah dari satu nilai ke nilai berikutnya mengikuti sinyal pemicu. Penghitung jumlah kendaraan, jam digital, penghitung skor, dan alat ukur frekuensi memakai prinsip ini. Setiap perubahan nilai harus disimpan sebelum hitungan berikutnya berlangsung. Di mana peran latch berada dalam proses tersebut? Elemen memori mempertahankan pola biner yang mewakili hitungan sekarang. Lo11 Hosea Pratama Sinaga

gika kombinasional menghitung keadaan berikutnya, lalu hasilnya disimpan kembali. Sebagai contoh, pencacah 2 bit bergerak dari 00 menuju 01, lalu 10, 11, dan kembali ke 00. Rangkaian perlu mengingat bahwa keadaan saat ini adalah 10 agar langkah berikutnya dapat ditentukan sebagai 11. Tanpa elemen penyimpanan, pola yang sudah terbentuk langsung hilang dan urutan tidak dapat dijaga. Latch memberi dasar untuk memahami proses tersebut walaupun counter praktis lebih sering memakai flip-flop yang dikendalikan oleh clock. Siswa yang sudah memahami keadaan sekarang dan keadaan berikutnya akan lebih mudah membaca urutan hitungan karena setiap pola tidak dianggap sebagai angka yang berdiri sendiri, melainkan bagian dari perjalanan sistem (Floyd, 2015; Tocci et al., 2017).

d. Peran Waktu dalam Kerja Latch Rangkaian digital memerlukan waktu singkat untuk merespons perubahan masukan. Saat Enable diaktifkan dan D berubah, keluaran Q tidak berganti pada saat yang benar-benar bersamaan. Sinyal harus melewati gerbang sebelum keadaan baru muncul. Selang tersebut disebut propagation delay. Pada percobaan sederhana, waktunya terlalu singkat untuk dilihat dengan mata. Pada sistem berkecepatan tinggi, perbedaan beberapa nanodetik dapat menentukan apakah data tersimpan dengan benar. Kapan nilai D harus dibuat stabil? Data perlu berada pada keadaan yang benar sebelum latch berhenti menerima masukan dan tetap stabil sesaat setelahnya. Waktu minimum sebelum peristiwa penyimpanan disebut setup time, sedangkan waktu minimum setelahnya disebut hold time. Istilah hold time berbeda dari kondisi hold pada tabel latch. Kondisi hold berarti keluaran mempertahankan data, sedangkan hold time adalah syarat waktu agar data berhasil disimpan. Jika masukan berubah terlalu dekat dengan batas penutupan Enable, rangkaian dapat menerima nilai yang tidak sesuai. Pembahasan ini menghubungkan tabel kebenaran yang terlihat pasti dengan sifat fisik komponen yang tetap membutuhkan waktu

untuk bekerja (Brown & Vranesic, 2014; Katz & Borriello, 2005).

e. Metastabilitas Latch dirancang untuk menyimpan keadaan 0 atau 1, tetapi ada kemungkinan singkat ketika keluarannya belum menetap pada salah satu nilai tersebut. Keadaan sementara ini disebut metastabilitas. Kondisi itu dapat muncul ketika data berubah terlalu dekat dengan waktu latch berhenti menerima masukan atau ketika dua perintah yang bertentangan dilepas hampir bersamaan. Mengapa keluarannya tidak langsung memilih 0 atau 1? Dua jalur internal dapat saling bersaing karena waktu respons setiap gerbang tidak pernah benar-benar identik. Selisih yang sangat kecil menentukan arah akhir rangkaian. Pada SR Latch, keadaan invalid dapat meningkatkan risiko hasil yang sulit diprediksi ketika Set dan Reset kembali tidak aktif pada waktu yang Hosea Pratama Sinaga 12

hampir sama. Materi sumber menyebut kombinasi S = 1 dan R = 1 sebagai kondisi yang tidak dapat dipastikan, lalu D Latch dipakai untuk menghindari kombinasi tersebut. Metastabilitas tidak berarti rangkaian rusak permanen. Keluaran biasanya akan menetap setelah waktu tertentu, tetapi keterlambatan itu dapat mengganggu bagian lain yang sudah menunggu data. Sistem digital memakai perancangan waktu, sinkronisasi, dan susunan elemen tambahan untuk mengurangi peluang masalah tersebut (Ginosar, 2011; Kinniment, 2007).

f. Latch, Flip-Flop, dan Sinyal Clock Latch bekerja selama Enable berada pada tingkat aktif, sedangkan flip-flop umumnya membaca data pada tepi clock. Clock adalah sinyal yang berubah berulang antara 0 dan 1 untuk mengatur kapan bagian-bagian sistem boleh memperbarui keadaannya. Mengapa komputer membutuhkan irama bersama? Banyak rangkaian bekerja dalam satu sistem dan harus memindahkan data tanpa saling mendahului. Clock memberi titik waktu yang disepakati untuk menerima atau mengirim keadaan baru. Latch dapat digunakan dalam rancangan yang membagi satu siklus clock menjadi beberapa fase. Data bergerak saat latch pertama terbuka, lalu ditahan ketika latch berikutnya menerima data. Susunan dua latch juga dapat membentuk flip-flop masterslave. Latch pertama menangkap data pada satu tingkat clock, latch kedua meneruskannya pada tingkat yang berlawanan. Keluaran akhirnya hanya berubah pada bagian tertentu dari siklus. Materi sumber membedakan latch yang dipicu oleh tingkat Enable dari flip-flop yang merespons perubahan tepi sinyal. Perbedaan ini membantu siswa memahami bahwa penyimpanan data berkaitan erat dengan pengaturan waktu, bukan hanya hubungan antarinput dan output (Givone, 2003; Holdsworth & Woods, 2002).

g. SR Latch untuk Mengatasi Pantulan Tombol Tombol mekanis tidak selalu menghasilkan satu perubahan sinyal yang bersih saat ditekan. Dua permukaan logam di dalam tombol dapat memantul beberapa kali sebelum benar-benar bersentuhan. Satu tekanan yang terlihat oleh manusia dapat terbaca sebagai beberapa pulsa oleh rangkaian digital. Gejala tersebut disebut

switch bounce. Bagaimana latch membantu mengatasinya? SR Latch dapat dipakai sebagai rangkaian debouncer untuk menyimpan satu keadaan setelah kontak pertama terdeteksi. Pantulan berikutnya tidak terus mengubah keluaran karena latch sudah berada pada keadaan set atau reset. Teknik ini berguna pada tombol penghitung, sakelar pilihan, dan masukan manual yang membutuhkan satu respons untuk setiap tindakan. Siswa dapat membayangkan sebuah counter yang seharusnya bertambah satu, tetapi justru meloncat beberapa angka karena tombol memantul. Latch menahan hasil agar satu penekanan hanya dianggap sebagai satu peristiwa. Contoh 13 Hosea Pratama Sinaga

ini membuat fungsi memori lebih dekat dengan penggunaan nyata. Latch bukan sekadar rangkaian di dalam prosesor, tetapi juga dapat menjadi solusi sederhana untuk merapikan sinyal dari komponen mekanis (Horowitz & Hill, 2015; Platt, 2012).

h. Latch pada Sistem Kendali Banyak mesin perlu mempertahankan sebuah perintah meskipun tombol hanya ditekan sebentar. Tombol start pada sistem kendali dapat memberi sinyal sesaat, lalu rangkaian menyimpan keadaan aktif sampai tombol stop ditekan. Prinsip tersebut serupa dengan fungsi Set dan Reset. Set memulai keadaan, Reset menghentikannya, dan hold menjaga keadaan di antara kedua perintah. Di mana pola ini dapat ditemukan? Sistem alarm, panel kontrol, lampu indikator, mesin produksi, dan rangkaian pengunci menggunakan gagasan yang sejenis. Sebuah alarm dapat diset ketika sensor mendeteksi gangguan, kemudian tetap menyala walaupun sinyal sensor sudah hilang. Alarm baru berhenti setelah pengguna memberikan perintah reset. Fungsi ini membuat peristiwa singkat tetap tercatat sehingga tidak terlewat. Pada praktik industri, kebutuhan keamanan dan karakter perangkat keras harus diperhitungkan lebih luas, tetapi model SR Latch sudah cukup untuk menjelaskan logika penyimpanan perintah. Siswa dapat mempelajari bahwa suatu sistem tidak selalu harus mengikuti masukan secara langsung. Kadang-kadang keadaan perlu dipertahankan sampai ada perintah yang secara khusus mengubahnya (Bolton, 2015; Petruzella, 2010).

i. Simulasi sebagai Cara Mengamati Memori Digital Proses penyimpanan satu bit berlangsung di dalam hubungan antargerbang yang sulit diamati hanya dari gambar. Simulator memberi kesempatan untuk mengubah masukan satu per satu, menelusuri jalur sinyal, dan melihat keluaran setelah perintah dilepas. Pengujian dapat dimulai dengan membuat Q bernilai 1 melalui Set. Setelah Set dikembalikan ke 0, siswa memeriksa apakah Q tetap bertahan. Percobaan dilanjutkan memakai Reset, kondisi hold, lalu kombinasi invalid. Pada D Latch, siswa dapat mengubah D ketika Enable aktif, kemudian menutup Enable dan mencoba mengganti D kembali. Apa yang harus dicatat agar simulasi tidak berubah menjadi kegiatan menekan tombol tanpa tujuan? Prediksi, keadaan awal, kombinasi masukan, keluaran, dan

penjelasan singkat perlu dimasukkan ke tabel pengamatan. Materi sumber memang meminta pembuatan SR Latch dan D Latch pada simulator serta pembuktian tabel kebenarannya. Simulasi memberi hasil belajar yang lebih kuat saat siswa dibimbing untuk membuat dugaan, menguji setiap kondisi, dan menjelaskan perbedaan antara prediksi dengan keluaran yang diamati (de Jong & van Joolingen, 1998; Rutten et al., 2012). Hosea Pratama Sinaga 14

j. Latch sebagai Jalan Menuju Sistem Digital yang Lebih Besar Pemahaman latch membuka jalan menuju materi yang lebih luas. D Latch dapat berkembang menjadi D Flip-Flop, beberapa flip-flop membentuk register, register dapat disusun menjadi pencacah, dan kumpulan keadaan dapat digunakan untuk membuat finite state machine. Mesin keadaan dipakai untuk mengatur urutan kerja perangkat, seperti mesin penjual otomatis, lampu lalu lintas, pengunci digital, dan protokol komunikasi. Setiap sistem memiliki keadaan sekarang, masukan, aturan perpindahan, serta keadaan berikutnya. Bagaimana sistem yang rumit dapat dibangun dari rangkaian kecil? Setiap bagian menyelesaikan satu tugas yang jelas. Latch menyimpan keadaan, gerbang logika menentukan perubahan, dan sinyal kendali mengatur kapan perubahan diterima. Dari susunan tersebut, perangkat dapat mengikuti urutan yang panjang tanpa kehilangan informasi mengenai langkah terakhirnya. Bagi siswa, latch menjadi titik awal untuk melihat komputer sebagai mesin yang memiliki keadaan internal, bukan sekadar kumpulan gerbang yang bereaksi sesaat. Satu bit memori memberi konsep dasar untuk membaca penyimpanan data, urutan proses, pengambilan keputusan, dan koordinasi waktu dalam elektronika digital (Nisan & Schocken, 2005; Patterson & Hennessy, 2021).

Penutup Rangkaian latch memperlihatkan titik ketika elektronika digital mulai memiliki kemampuan mengingat. Gerbang logika yang awalnya hanya mengolah masukan saat ini dapat mempertahankan hasil sebelumnya melalui jalur umpan balik. Satu keluaran Q sudah cukup untuk menyimpan satu bit, sedangkan gabungan banyak elemen memori dapat membentuk register dan sistem penyimpanan yang lebih besar. Apa yang sebenarnya dipelajari siswa dari rangkaian sekecil ini? Mereka belajar bahwa komputer tidak memahami ingatan seperti manusia. Komputer mempertahankan pola 0 dan 1 di dalam susunan rangkaian yang memiliki keadaan stabil. SR Latch memperkenalkan perintah set, reset, hold, dan kondisi invalid. D Latch menyederhanakan proses penulisan data melalui satu masukan D yang dikendalikan oleh Enable. Urutan tersebut memberi jalur belajar yang jelas dari gerbang logika menuju sistem digital yang menyimpan dan memindahkan informasi. Materi komputasi tingkat sekolah juga menempatkan perangkat keras, data, dan cara kerja sistem sebagai bagian dasar untuk memahami teknologi secara utuh (National Academies of Sciences, Engineering, and Medicine, 2018; Computer Science Teachers Asso-

ciation, 2017). Pembelajaran latch akan lebih mudah diterima ketika siswa ikut mengubah masukan dan mengamati perubahan keluaran. Simulasi dapat menampilkan jalur sinyal, kondisi Enable, keluaran Q, serta $\bar{Q}$ tanpa menunggu penyusunan perangkat keras yang rumit. Rangkaian fisik tetap berguna untuk memper15 Hosea Pratama Sinaga

kenalkan kabel, catu daya, gerbang logika, dan kesalahan pemasangan yang muncul dalam percobaan nyata. Bagaimana siswa dapat membedakan proses menyimpan data dari keluaran yang sekadar mengikuti masukan? Mereka dapat menulis nilai awal Q, mengaktifkan Set atau memasukkan nilai D, menonaktifkan Enable, lalu mengubah masukannya kembali. Keluaran yang tetap bertahan menjadi bukti bahwa rangkaian sedang menyimpan keadaan sebelumnya. Prediksi sebelum percobaan, pencatatan hasil, dan pembahasan kesalahan membuat simulasi memiliki arah yang jelas. Kajian mengenai simulasi komputer mencatat manfaatnya dalam pembelajaran konsep sains, sedangkan pendidikan STEM terintegrasi mendorong kegiatan yang menghubungkan pengetahuan, percobaan, dan perancangan dalam satu pengalaman belajar (Rutten et al., 2012; National Academy of Engineering & National Research Council, 2014). Pemahaman latch dapat dilanjutkan menuju flip-flop, register, pencacah, mesin keadaan, dan memori digital. Setiap materi tersebut masih memakai gagasan yang sama, yaitu menyimpan keadaan sekarang agar rangkaian dapat menentukan tindakan berikutnya. Siswa tidak perlu memulai dari arsitektur komputer yang besar. Dua gerbang yang saling terhubung sudah dapat membuka pembahasan mengenai data, waktu, urutan, dan pengendalian sistem. Kapan pembelajaran dapat disebut berhasil? Ukurannya terlihat ketika siswa mampu memperkirakan keluaran, menjelaskan alasan nilai Q bertahan, membedakan kondisi set dan reset, serta membuktikan tabel keadaan melalui percobaan sendiri. Materi sumber juga menugaskan pembuatan SR Latch dan D Latch pada simulator beserta pembuktian tabel kebenarannya. Kegiatan eksplorasi seperti ini perlu dilengkapi arahan dan pertanyaan pengamatan agar siswa tidak berhenti pada kegiatan menekan sakelar tanpa memahami model rangkaiannya (Modul Rangkaian Latch, n.d.; de Jong & van Joolingen, 1998).

Kesimpulan Rangkaian latch merupakan elemen memori dasar pada elektronika digital yang mampu menyimpan satu bit informasi dalam bentuk 0 atau 1. Cara kerjanya berbeda dari rangkaian kombinasional karena keluaran latch dipengaruhi oleh masukan saat ini dan keadaan keluaran sebelumnya. Jalur umpan balik membuat nilai lama tetap bertahan sampai ada perintah baru yang mengubahnya. Prinsip ini menjadi dasar bagi rangkaian sekuensial, register, pencacah, flip-flop, dan berbagai sistem penyimpanan digital. SR Latch menggunakan masukan Set, Reset, dan Enable. Set membuat Q bernilai 1, Reset membuat Q bernilai 0, sedangkan kombinasi Set dan Reset bernilai 0 mempertahankan keadaan sebelumnya. Kondisi Set

dan Reset yang aktif bersamaan menghasilkan keadaan invalid karena keluaran berikutnya sulit dipastikan. D Latch memakai satu masukan data untuk menghindari Hosea Pratama Sinaga 16

kondisi tersebut. Ketika Enable bernilai 1, keluaran mengikuti nilai D. Ketika Enable bernilai 0, keluaran menyimpan nilai terakhir yang telah diterima. Simulasi SR Latch dan D Latch membantu siswa membuktikan tabel kebenaran, membaca perubahan keluaran, dan memahami proses penyimpanan data secara langsung. Rangkaian yang hanya terdiri dari beberapa gerbang ini memberi gambaran awal tentang cara komputer mempertahankan informasi dan menggunakan keadaan sebelumnya untuk menentukan proses berikutnya.

Daftar Pustaka

Balabanian, N., & Carlson, B. S. (2001). Digital logic design principles. John Wiley & Sons.

Bolton, W. (2015). Mechatronics: Electronic control systems in mechanical and electrical engineering (6th ed.). Pearson.

Brown, S., & Vranesic, Z. (2014). Fundamentals of digital logic with VHDL design (3rd ed.). McGraw-Hill Education.

Comer, D. J. (1995). Digital logic and state machine design (3rd ed.). Oxford University Press. Computer Science Teachers Association. (2017). CSTA K-12 computer science standards: Revised 2017. https://csteachers.org/2017standards/ de Jong, T., & van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. Review of Educational Research, 68(2), 179-201. https://doi.org/10.3102/00346543068002179

Even, G., & Medina, M. (2012). Digital logic design: A rigorous approach. Cambridge University Press.

Floyd, T. L. (2015). Digital fundamentals (11th ed.). Pearson.

Ginosar, R. (2011). Metastability and synchronizers: A tutorial. IEEE Design & Test of Computers, 28(5), 23-35.

Givone, D. D. (2003). Digital principles and design. McGraw-Hill.

Harris, D. M., & Harris, S. L. (2012). Digital design and computer architecture (2nd ed.). Morgan Kaufmann.

Hayes, J. P. (1993). Introduction to digital logic design. Addison-Wesley.

Holdsworth, B., & Woods, R. C. (2002). Digital logic design (4th ed.). Newnes.

Horowitz, P., & Hill, W. (2015). The art of electronics (3rd ed.). Cambridge University Press.

Katz, R. H., & Borriello, G. (2005). Contemporary logic design (2nd ed.). Pearson Prentice Hall.

Kinniment, D. J. (2007). Synchronization and arbitration in digital systems. John Wiley & Sons. https://doi.org/10.1002/9780470517147 17 Hosea

Pratama Sinaga
Kleitz, W. (2012). Digital electronics: A practical approach with VHDL (9th ed.). Pearson.
Lala, P. K. (2007). Principles of modern digital design. John Wiley & Sons. https: //doi.org/10.1002/0470125217
Mano, M. M., & Ciletti, M. D. (2018). Digital design: With an introduction to the Verilog HDL, VHDL, and SystemVerilog (6th ed.). Pearson.
Mano, M. M., & Kime, C. R. (2008). Logic and computer design fundamentals (4th ed.). Pearson Prentice Hall.
Nahin, P. J. (2017). The logician and the engineer: How George Boole and Claude Shannon created the information age. Princeton University Press. National Academies of Sciences, Engineering, and Medicine. (2018). How people learn II: Learners, contexts, and cultures. National Academies Press. https://doi. org/10.17226/24783 National Academy of Engineering, & National Research Council. (2014). STEM integration in K-12 education: Status, prospects, and an agenda for research. National Academies Press. https://doi.org/10.17226/18612
Nelson, V. P., Nagle, H. T., Carroll, B. D., & Irwin, J. D. (1995). Digital logic circuit analysis and design. Prentice Hall.
Nisan, N., & Schocken, S. (2005). The elements of computing systems: Building a modern computer from first principles. MIT Press.
Nisan, N., & Schocken, S. (2021). The elements of computing systems: Building a modern computer from first principles (2nd ed.). MIT Press.
Patterson, D. A., & Hennessy, J. L. (2021). Computer organization and design: The hardware/software interface (6th ed.). Morgan Kaufmann.
Petruzella, F. D. (2010). Programmable logic controllers (4th ed.). McGraw-Hill.
Platt, C. (2015). Make: Electronics: Learning through discovery (2nd ed.). Maker Media. Rangkaian latch. (n.d.). [Modul pembelajaran elektronika digital].
Roth, C. H., Jr., & Kinney, L. L. (2014). Fundamentals of logic design (7th ed.). Cengage Learning.
Rutten, N., van Joolingen, W. R., & van der Veen, J. T. (2012). The learning effects of computer simulations in science education. Computers & Education, 58(1), 136-153. https://doi.org/10.1016/j.compedu.2011.07.017
Tocci, R. J., Widmer, N. S., & Moss, G. L. (2017). Digital systems: Principles and applications (12th ed.). Pearson.
Wakerly, J. F. (2006). Digital design: Principles and practices (4th ed.). Pearson Prentice Hall.